$ >

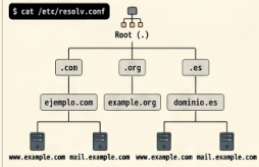
Manual Técnico: Servidor DNS y Administración de Sistemas

DRAFT

Autor: Arcadio Ortega Reinoso

ARQUITECTURA Y HERRAMIENTAS DE DIAGNÓSTICO

Jerarquía y Estructura DNS



Operación jerárquica desde Root (.) hasta TLDs, SLDs y Hosts específicos.

Flujo de Resolución Local

1. /etc/hosts Prioridad alta
2. Caché Local Systemd-resolved
3. Servidores DNS Configurado en /etc/resolv.conf

Comandos Esenciales

Dig (Análisis detallado)

```
$ dig +trace example.com
$ ... ... ...
RRSIG    A   example.com
RRSIG    A   example.com
RRSIG    A   example.com
RRSIG    A   example.com
```

Análisis detallado de registros y trazado.

Resolvectl (Gestión systemd-resolved)

```
$ resolvectl status

Global: ...
Link 2 (ofn8): ...
  DNSSEC=allow-downgrade
...
```

Gestionar systemd-resolved y estado de enlaces.

SEGURIDAD AVANZADA Y PROTOCOLOS

Privacidad: DoT vs. DoH

DoT (DNS sobre TLS) Cifrado de consultas, conexión persistente.

DoH (DNS sobre HTTPS) Cifrado sobre protocolo web estándar.

Integridad con DNSSEC

Añade firmas digitales (RRSIG) para prevenir envenenamiento de caché y suplantación.

Mecanismos de Control (TSIG y RRL)

Autenticación de transferencias de zona.

Mitigación de ataques DoS limitando respuestas.

COMPARATIVA DE SERVIDORES

$ cat /etc/dns_server_comparison.txt

Características	BIND9	PowerDNS	dnsmasq
Complejidad	Alta	Media	Baja
Escalabilidad	Muy Alta	Muy Alta	Baja
Ideal para	Grandes Empresas	ISPs / DB Modernas	Redes Pequeñas

NotebookLM

Bloques

Bloque 1: Fundamentos Conceptuales e Introducción Teórica

El manual arranca planteando el problema clásico de la comunicación en redes: la brecha natural entre humanos (que recordamos nombres) y máquinas (que usan IPs), introduciendo el DNS bajo la analogía de una guía telefónica. Explica de forma jerárquica las fases de una consulta (Resolutor, Servidores Raíz, TLD y Servidores Autorizados) y la optimización de velocidad mediante la memoria caché. Posteriormente, desglosa la anatomía formal del espacio de nombres (TLD, SLD, subdominios frente a hosts, la regla de especificidad de izquierda a derecha) y define el FQDN (Fully Qualified Domain Name), cerrando con los conceptos de servidor autoritativo y archivo de zona.

Bloque 2: Estructura de Datos y Mecánica Interna

Entra en el detalle de la información con el registro SOA (Start of Authority), diseccionando sus parámetros críticos: el Serial Number (explicando el formato recomendado YYYYMMDDNN) y los temporizadores de Refresh, Retry, Expire y Minimum TTL, vitales para coordinar servidores secundarios. A continuación, cataloga la tipología de registros comunes (A, AAAA, CNAME, MX, TXT, PTR, NS). Introduce además el rol de `systemd-resolved` (127.0.0.53) en distribuciones Linux modernas y esquematiza visualmente "El Viaje de una Consulta" en una resolución recursiva.

Bloque 3: Configuración Práctica e Implementación con BIND9

Da comienzo la fase de administración práctica en servidores a

través de BIND9. Se presenta la sintaxis de `named.conf` y sus directivas principales dentro del bloque `options` (directorio de almacenamiento, habilitación de la recursión, directivas `allow-query` y los `forwarders` apuntando a DNS públicos, además del comportamiento de `auth-nxdomain` bajo el estándar RFC1035). Culmina enseñando a maquetar y corregir un archivo de zona maestro real (`db.example.com`), aplicando la lógica de la directiva `$TTL` y el uso del símbolo "@".

Bloque 4: Entornos Multiservidor y Herramientas de Auditoría

El manual expande la infraestructura enseñando el modelo Maestro-Esclavo (servidores secundarios), detallando directivas avanzadas de BIND como las listas de control de acceso (`acl "trusted"`) y la restricción de transferencias de zona mediante `allow-transfer`. En paralelo, capacita en herramientas críticas de diagnóstico en vivo: el uso de `dig` para examinar banderas de respuesta y el uso de `tcpdump` para la captura de paquetes en interfaces específicas (remarcando el uso crítico del parámetro `-n` para evitar bucles infinitos de resolución). Introduce también el control del daemon en caliente mediante `rndc` (Remote Name Daemon Control) para recargar zonas (`rndc reload`), limpiar la caché (`rndc flush`) o auditar estados (`rndc status`).

Bloque 5: Seguridad Avanzada, Cifrado y Privacidad

El texto se orienta firmemente hacia el *hardening* y la protección del servicio. Aborda la mitigación de ataques de amplificación DNS limitando la recursión y ocultando la versión del software de las miradas de posibles atacantes. Introduce detalladamente **DNSSEC** de manera educativa y manual, explicando las firmas RRSIG, los registros DS y el diagnóstico de fallos criptográficos (como el estado "BOGUS" y el error SERVFAIL). Compara esquemáticamente las tecnologías de privacidad **DoT** (DNS over TLS en puerto 853) y **DoH** (DNS over HTTPS en puerto 443), y describe otros mecanismos como TSIG para transferencias seguras, Response Rate Limiting (RRL) y DANE.

Bloque 6: Alternativas Tecnológicas y Zonas Especiales

El libro realiza una pausa en BIND9 para ofrecer una perspectiva comparativa del ecosistema de software alternativo según las necesidades del entorno: **PowerDNS** (modular, optimizado para ISPs y bases de datos SQL), **dnsmasq** (ligero, ideal para redes domésticas o routers que requieren caché y DHCP integrado sin la complejidad de DNSSEC), junto a soluciones como **djbdns**, **NSD**, **MaraDNS** y **Pi-hole** (como bloqueador de publicidad en la red local). También introduce el tratamiento de archivos de zonas especiales para redes privadas como `/etc/bind/zones.rfc1918`.

Bloque 7: Entornos Locales, Descubrimiento de Servicios y Cierre

El manual concluye aterrizando en la periferia de la red local: el descubrimiento de servicios sin infraestructura centralizada a través de **mDNS** y **DNS-SD** (Zeroconf/Avahi). Contrasta los riesgos de seguridad de mDNS local frente a Wide-Area DNS-SD protegido por DNSSEC. El broche de oro son guías prácticas y comandos explícitos para desactivar mDNS y LLMNR tanto en Linux (modificando `resolved.conf` en `systemd-resolved`) como en Windows (desactivando el servicio de anuncios UPnP / SSDPSRV y configurando directivas con `gpedit.msc`).

DNS. Estructura del manual

La estructura del manual busca una **curva de aprendizaje progresiva, acumulativa y estrictamente lógica**:

1. **Qué es y cómo se organiza** (Concepto macro).
2. **Qué datos viajan por él** (Registros de recursos).
3. **Cómo se monta el servidor base** (Maestro).
4. **Cómo se escala y se vigila** (Esclavo y Diagnóstico con dig/ tcpdump/rndc).
5. **Cómo se blinda** (Seguridad avanzada y Cifrado con DNSSEC/DoH/DoT).

6. **Qué otras herramientas existen** (Alternativas de software).
7. **Cómo interactúa mi propio PC local y cómo fortificarlo** (mDNS/LLMNR y Endpoints).

Segunda parte

Este manual tendrá una continuación en una segunda parte orientada a la seguridad.

Aunque se lleve el grueso de la seguridad a una segunda parte aquí tratamos:

- Fundamentos de Seguridad: El marco teórico y el control de acceso inicial (ACLs, restricción de recursión).
- Criptografía e Integridad : Autenticación de zonas.
- Privacidad del Canal: Cifrado del tráfico (DoT).
- Mitigación y Filtrado Operativo: Bloqueo de amenazas (RPZ/ Listas negras).
- El subbloque de Redes Locales: mDNS, dns-sd, avahi y llmnr, para terminar con un capítulo orientado a desactivar todos estos sistemas

Lección: El Sistema de Nombres de Dominio (DNS) - La Guía Telefónica de Internet

1. El Problema: Humanos vs. Máquinas

En el mundo de las redes, existe una "brecha de comunicación" natural. Las **computadoras y dispositivos se identifican mediante números**, específicamente **direcciones IP**, porque así es como "hablan" entre sí. Por el contrario, los **seres humanos estamos acostumbrados a usar nombres** para identificar lugares o cosas, ya que nos resulta casi imposible memorizar las direcciones IP de los millones de sitios web que existen en Internet.

Para solucionar esto, los ingenieros desarrollaron el **DNS (Domain Name System)**, que actúa básicamente como una **guía telefónica**. Cuando quieres llamar a alguien, buscas su nombre y la guía te da su número; el DNS hace lo mismo: tú le das un nombre de dominio (como yahoo.com) y él te devuelve la dirección IP necesaria para que tu computadora se conecte al servidor web.

2. El Proceso de Consulta: ¿Qué pasa cuando escribes una URL?

Cuando introduces una dirección en tu navegador, se inicia una búsqueda organizada por niveles. Si tu propia computadora no tiene la dirección guardada en su memoria interna (caché), el proceso sigue estos pasos:

1. **El Resolver (Resolutor):** Es el primer nivel de consulta fuera de tu equipo, generalmente proporcionado por tu **ISP (Proveedor de Servicios de Internet)**. El resolutor revisa su propia memoria; si no tiene la respuesta, escala la pregunta al siguiente nivel.
2. **Servidores Raíz (Root Servers):** Son la cima de la jerarquía

mundial. Existen **13 conjuntos de estos servidores** distribuidos por todo el planeta. Aunque el servidor raíz no sabe la IP exacta que buscas, **sabe a quién preguntarle**. Dirigirá al resolutor hacia el servidor encargado del dominio de nivel superior (TLD) correspondiente.
3. **Servidores TLD (Top-Level Domain):** Estos servidores gestionan extensiones específicas como **.com, .net o .org**. Si buscas un ".com", el servidor TLD te enviará al último nivel de la cadena.
4. **Servidores de Nombres Autorizados (Authoritative Name Servers):** Son la **autoridad final**. Estos servidores tienen toda la información sobre el dominio, incluyendo su dirección IP definitiva. Una vez que el servidor autorizado responde, el resolutor recibe la IP y se la entrega a tu computadora.

3. Eficiencia y Velocidad: La Memoria Caché

Para evitar que este viaje por todo el mundo se repita cada vez que entras a la misma página, el **resolutor almacena la dirección IP en su memoria caché**. De esta forma, si tú u otro usuario vuelve a consultar el mismo dominio, el resolutor puede responder inmediatamente sin tener que pasar por los servidores raíz o TLD de nuevo, ahorrando tiempo y recursos de red.

La Jerarquía del DNS: De lo Específico a lo General

El sistema DNS se organiza mediante una estructura jerárquica descentralizada. Para entenderla, debemos leer los dominios de **izquierda a derecha**:

- **TLD (Top-Level Domain):** Son la base de la pirámide (como `.com` o `.edu`).
- **SLD (Second-Level Domain):** Es el nombre que contratamos bajo un TLD, como `ubuntu` en `ubuntu.com`. Técnicamente, el SLD es un subdominio del TLD.
- **Subdominios vs. Hosts:** Es vital distinguir estos conceptos.

Un **subdominio** es una subdivisión lógica que extiende al dominio padre para organizar la red (ej. `historia` en `historia.escuela.edu`). Por el contrario, un **nombre de host** identifica un recurso o máquina específica (como el servidor `www`).

- **La Regla de Especificidad:** Al leer una dirección, la parte situada más a la izquierda es siempre la más específica (el host o equipo), y nos movemos hacia la menos específica (el TLD) conforme avanzamos a la derecha.

Esta organización permite que cada nivel de la jerarquía tenga control total sobre los subdominios que se encuentran debajo de él, facilitando una administración escalable y ordenada.

Terminología

Domain Name System

El sistema de nombres de dominio, más comúnmente conocido como "DNS", nos permite resolver nombres entendibles por el ser humano en lugar de usar direcciones IP.
La relación entre ellos es biunívoca
El DNS es el traductor universal de Internet que convierte los nombres que nosotros entendemos en los números que las máquinas necesitan, utilizando una estructura jerárquica que garantiza que siempre encontremos el destino correcto.

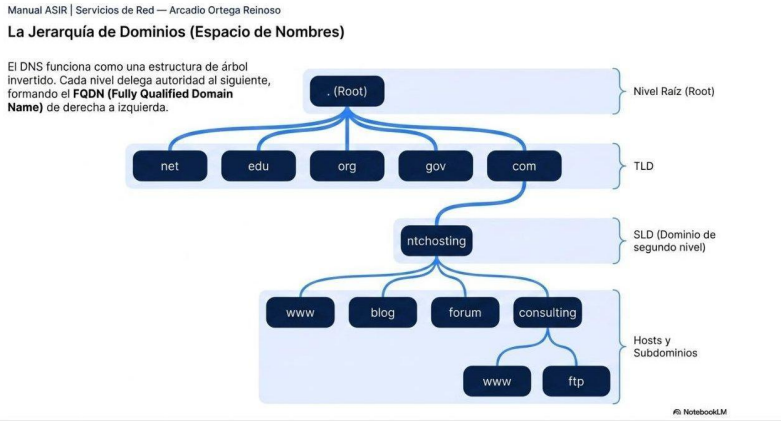
Domain Name

Un nombre de dominio es el nombre que comprendemos y asociamos con un recurso de internet.
Por ejemplo,google.com Es un nombre de dominio.
Algunas personas dirán que la porción de google es el dominio, pero generalmente podemos referirnos a la forma combinada como el nombre de dominio. La URL google.com Está asociado con los servidores propiedad de Google Inc.
El sistema de nombres de dominio nos permite llegar a los

servidores de Google cuando tecleamos www..google.com en nuestros navegadores.

Top-Level Domain (Dominio de primer nivel)

jerarquía



Un dominio de nivel superior, o **TLD**, es la parte más general del dominio.
El dominio de nivel superior es la porción más lejana a la derecha (como se separa de un punto).
Los dominios comunes de alto nivel son los de .com, .net, .org, .gov,...

Los dominios de nivel superior están en la parte superior de la jerarquía en términos de nombres de dominio.
Algunas partes reciben el control de gestión de los dominios de alto nivel por parte de ICANN (Internet Corporation for Assigned Names and Numbers).
Estas partes pueden distribuir nombres de dominio bajo el TLD, generalmente a través de un registrador de dominios.

Hosts

Dentro de un dominio, el propietario del dominio puede definir hosts individuales, que se refieren a computadoras o servicios separados accesibles a través de un dominio.
Por ejemplo, la mayoría de los propietarios de dominios hacen que

sus servidores web sean accesibles a través del dominio example.com y también a través de la definición de "host" www (www.example.com).
Podría tener acceso API a través de un host de Api (api.example.com) o podrías tener acceso ftp definiendo un host llamado "ftp" o "archivo" (ftp.example.como o files.example.com).
Los nombres de host pueden ser arbitrarios siempre y cuando sean únicos para el dominio.

Subdominio

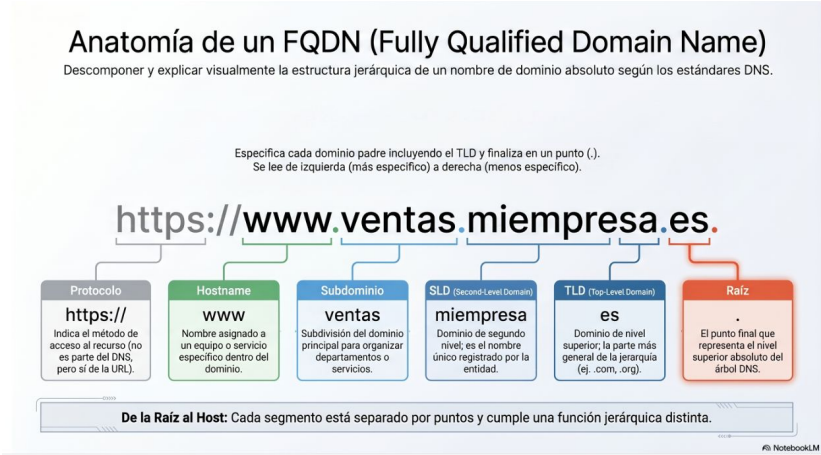
DNS trabaja en una jerarquía.
Un "subdominio" se refiere a cualquier dominio que sea parte de un dominio más grande.
Por ejemplo: ubuntu.com Se puede decir que es un subdominio de com. La porción de "ubuntu" se llama **SLD**, que significa dominio de segundo nivel.
Del mismo modo, cada dominio puede controlar los subdominios que se encuentran debajo de él.

> *La diferencia entre un nombre de host y un subdominio es que un host define una computadora o recurso, mientras que un subdominio extiende el dominio padre.*

Ya sea hablando de subdominios o anfitriones, puede empezar a ver que las porciones más izquierda de un dominio son las más específicas.
Así es como funciona DNS: de la mayoría a la menos específica como lees de izquierda a derecha.

Fully Qualified Domain Name

Fully Qualified Domain Name



Un nombre de dominio totalmente calificado, a menudo llamado FQDN, es lo que llamamos un nombre de dominio absoluto.
Los dominios en el sistema DNS se pueden dar en relación entre si, y como tal, pueden ser algo ambiguos.
Un FQDN es un nombre absoluto que especifica su ubicación en relación con la raíz absoluta del sistema de nombres de dominio.
Esto significa que especifica cada dominio padre incluyendo el TLD.
Un FQDN adecuado termina con un punto, indicando la raíz de la jerarquía DNS.
Un ejemplo de FQDN es "mail.google.com.". A veces el software que llama FQDN no requiere el punto final, pero el punto de seguimiento se requiere para ajustarse a los estándares de ICANN.

Name Server

Un servidor de nombres es un ordenador designado para traducir los nombres de dominio en direcciones IP.
Estos servidores hacen la mayor parte del trabajo en el sistema DNS.
Dado que el número total de traducciones de dominio es demasiado para cualquier servidor, cada servidor puede redirir y dirigir la solicitud a otros servidores de nombres o delegar la responsabilidad de un subconjunto de subdominios de los que son responsables.

Autoritativos

Los servidores de nombres pueden ser "autoritativos", lo que significa que dan respuestas a las consultas sobre los dominios bajo su control.
De lo contrario, pueden apuntar a otros servidores, o servir copias en caché de otros servidores de nombres.

Zone File (Archivo de zona)

Un archivo de zona es un archivo de texto simple que contiene las asignaciones entre nombres de dominio y direcciones IP.
Así es como el sistema DNS finalmente descubre qué dirección IP debe ponerse en contacto cuando un usuario solicita un determinado nombre de dominio.
Los archivos de zona residen en servidores de nombres y generalmente definen los recursos disponibles bajo un dominio específico, o el lugar que uno puede ir para obtener esa información.

El Registro SOA: El Corazón de la Autoridad DNS

El registro **Start of Authority (SOA)** es el primer registro de recursos que debe aparecer en cualquier archivo de zona. Su función principal es declarar que ese servidor DNS es la **fuente de información autoritativa** para el dominio y establecer los parámetros de comunicación para los servidores secundarios.

1. El Número de Serie (Serial Number)

Es el identificador de la versión del archivo de zona. Es vital para la sincronización entre servidores Maestros y Esclavos:

- **Funcionamiento:** Cada vez que realices un cambio en los registros de `db.example.com`, debes incrementar este número manualmente.
- **Consecuencia de omitirlo:** Si el número de serie no cambia, los servidores secundarios (esclavos) asumirán que no hay actualizaciones y no sincronizarán los nuevos datos, provocando que la red propague información obsoleta.
- **Formato recomendado:** Se suele utilizar el formato `YYYYMMDDNN` (Año-Mes-Día-Revisión). Por ejemplo:

```
2026061701.
```

2. Tiempos de Refresco y Gestión de Errores

El registro SOA define cómo interactúan los servidores de la jerarquía DNS:

- **Refresh (Refresco):** Indica cada cuánto tiempo el servidor secundario debe preguntar al maestro si hay cambios en la zona.
- **Retry (Reintento):** Si el servidor maestro no responde, este valor define cuánto tiempo debe esperar el secundario antes de intentar una nueva conexión.
- **Expire (Expiración):** Si el servidor maestro permanece inaccesible durante este periodo, el secundario dejará de considerar válida su copia de la zona y dejará de responder consultas.
- **Minimum TTL:** Define el tiempo de vida por defecto para el almacenamiento en caché de los registros que no existen (caché negativa).

Registros

Dentro de un archivo de zona, los registros se mantienen.
En su forma más simple, un registro es básicamente un solo mapeo entre un recurso y un nombre.
Estos pueden mapear un nombre de dominio a una dirección IP, definir los servidores de nombres para el dominio, definir los servidores de correo para el dominio, etc.

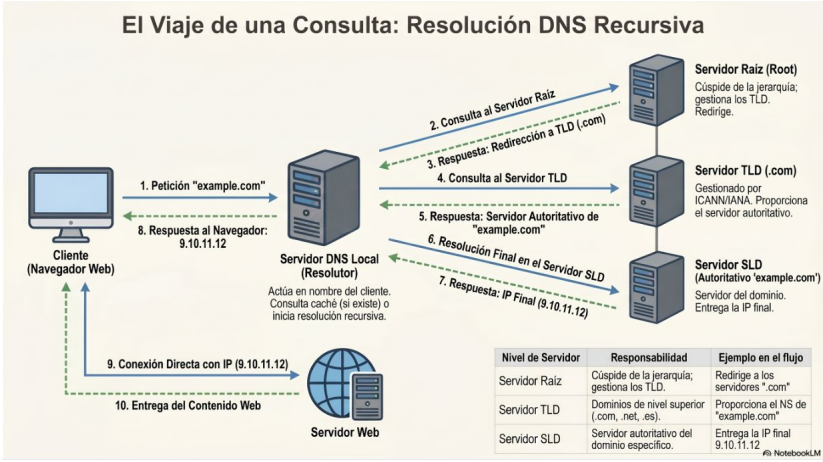
127.0.0.53

En muchos sistemas Linux modernos, especialmente aquellos que utilizan systemd como sistema de inicio, el servicio systemd-resolved actúa como un servidor DNS local y utiliza la dirección IP 127.0.0.53.(es la IP del stub resolver de systemd-resolved y no necesariamente el servidor DNS final)
systemd-resolved es un componente de systemd que proporciona servicios de resolución de nombres, incluida la resolución de DNS. Ayuda a centralizar la gestión de la resolución de nombres en el

sistema y puede proporcionar características adicionales, como la resolución de nombres locales y la gestión de caché

¿Cómo trabaja?

Cómo funciona DNS



DNS básico

```
options {
    directory "/var/cache/bind";
    recursion yes;
    allow-query { any; };
    forwarders {
        8.8.8.8;
        8.8.4.4;
    };
    //auth-nxdomain no;
    listen-on { any; };
};
```

directory "/var/cache/bind";

Especifica el directorio donde BIND almacenará sus datos, como las zonas y los registros de caché.

recursion yes;

Habilita la recursión, permitiendo que el servidor DNS resuelva consultas para cualquier cliente.

allow-query { any; };

Permite consultas desde cualquier dirección IP. Esto significa que el servidor responderá a consultas de cualquier lugar.

forwarders {...};

Especifica servidores DNS a los que se enviarán consultas que el servidor no puede resolver localmente. En este caso, se están utilizando los servidores DNS de Google (8.8.8.8 y 8.8.4.4).

auth-nxdomain no;

Cuando haces una consulta DNS para un nombre de dominio y el servidor no encuentra una entrada de registro correspondiente, responde con un código de respuesta NXDOMAIN. Esto indica que el nombre de dominio no existe en el contexto de la consulta realizada.

En este caso, se establece en "no" para conformarse con el estándar RFC1035.

Un "nombre de dominio no existente" es aquel para el cual no se ha configurado ninguna información de resolución en el sistema de nombres de dominio.

La recursividad se refiere a la capacidad de buscar activamente información en nombre del cliente, mientras que auth-nxdomain afecta a cómo se manejan las respuestas cuando un servidor DNS no es autoritativo para la zona consultada.

Opciones de auth-nxdomain:

auth-nxdomain yes; (Opción predeterminada)

Si auth-nxdomain está configurado en "yes," el servidor DNS solo responderá con NXDOMAIN si es autoritativo para la zona consultada. Esto significa que buscará en la zona autoritativa antes de responder con NXDOMAIN. Esta es la configuración predeterminada.

auth-nxdomain no;
: Si auth-nxdomain está configurado en "no," el servidor DNS responderá con NXDOMAIN de inmediato, incluso si no es autoritativo para la zona consultada. No buscará en la zona autoritativa antes de proporcionar la respuesta.

auth-nxdomain no; redirect;
: Esta opción se utiliza para redirigir las consultas de nombres de dominio no existentes a un servidor específico. Si el servidor DNS no es autoritativo para la zona, en lugar de proporcionar directamente la respuesta NXDOMAIN, redirigirá la consulta a otro servidor especificado.

Continuación de directivas:

listen-on { any; };
: Configura el servidor para escuchar en todas las interfaces de red. Puedes ajustar esto según tus necesidades específicas.

Ejemplo de un sistema básico:

```
  named.conf
options {
    directory "/var/named";
    allow-recursion { localhost; };
    allow-transfer { none; };
    recursion yes;
};

zone "example.com" IN {
    type master;
    file "db.example.com";
};
```

La recursividad en DNS implica que el servidor DNS puede realizar consultas en nombre de un cliente y buscar la respuesta incluso si no tiene la información en su propia zona. Esto es útil cuando el servidor DNS actúa como un resolutor para los clientes de la red.

- allow-recursion: Indica las direcciones IP que tienen permiso para realizar consultas recursivas al servidor DNS.

- { localhost; }: Especifica las direcciones IP permitidas para realizar consultas recursivas. Solo se permite la recursividad desde la dirección IP de localhost (127.0.0.1).

Solo el servidor DNS que se ejecuta en la misma máquina (localhost) tiene permitido realizar consultas recursivas al servidor. por lo que necesitaremos forwarders:

```
forwarders {
 8.8.8.8;
 8.8.4.4;
};
forward only;
```

forward only; indica que el servidor solo debe enviar consultas recursivas a los forwarders y no realizar búsquedas directas en la zona local.
IMPORTANTE: El servidor DNS solo enviará consultas recursivas a los forwarders especificados y no buscará localmente en las zonas que tenga configuradas.
Tampoco usaríamos: forward first;

```
  db.example.com
; Archivo de zona corregido para example.com
$TTL 86400
@   IN  SOA     ns1.example.com. admin.example.com. (
                2026061701  ; Serial
                3600        ; Refresh (1 hora)
                1800        ; Retry (30 mins)
                604800      ; Expire (1 semana)
                86400 )     ; Minimum TTL

; Registros NS y A definidos en el manual original
@   IN  NS      ns1.example.com.
@   IN  A       192.168.1.1
ns1 IN  A       192.168.1.1
www IN  A       192.168.1.2
```

Nota: "@" se utiliza para representar el dominio raíz o el dominio principal al que se hace referencia.
Quedando nuestro ejemplo: El servidor DNS resolverá consultas

para las zonas locales definidas y enviará consultas externas a los forwarders si no puede resolverlas localmente.

```
named.conf
options {
    directory "/var/named";
    allow-recursion { localhost; };

    forwarders {
        8.8.8.8;
        8.8.4.4;
    };
    forward only;
};

zone "example.com" IN {
    type master;
    file "db.example.com";
};
```

y el db.example.com como se propuso antes

Consultas recursivas

Una consulta recursiva se refiere a una solicitud de resolución de nombre de dominio en la que el servidor DNS al que se hace la consulta se compromete a proporcionar la respuesta completa al cliente.
Esto implica que, si el servidor DNS no tiene la información necesaria en su caché, realizará consultas adicionales a otros servidores DNS para obtener la información y luego responderá al cliente con la respuesta completa.

Cliente solicita una resolución de nombre de dominio
: Por ejemplo, un cliente solicita la resolución del nombre de dominio "www.ejemplo.com".

Servidor DNS recibe la solicitud
: El servidor DNS al que se envió la solicitud verifica si tiene la información en su caché. Si no la tiene, realizará una

serie de consultas recursivas para obtener la respuesta.

El servidor DNS consulta otros servidores

El servidor DNS realiza consultas a otros servidores DNS, siguiendo la cadena de autoridad desde la raíz del sistema de nombres de dominio hasta el servidor autoritario del dominio específico.

Obtiene la respuesta

El servidor DNS recopila la información necesaria para responder a la solicitud original y envía la respuesta completa al cliente.

Caché

La respuesta se almacena en la caché del servidor DNS para futuras consultas similares, lo que mejora la eficiencia y reduce la necesidad de realizar consultas recursivas nuevamente.

/etc/resolv.conf

El archivo /etc/resolv.conf en sistemas basados en Unix/Linux contiene la configuración de resolución de nombres para el sistema. Este archivo especifica los servidores DNS a los que el sistema debe consultar al realizar consultas de resolución de nombres.

nameserver 127.0.0.53:

Esta línea especifica la dirección IP del servidor DNS al que el sistema enviará consultas de resolución de nombres. Esta dirección IP es comúnmente utilizada por systemd-resolved

options edns0 trust-ad:

- edns0 se refiere a la extensión de DNS (EDNS) que permite a los clientes y servidores DNS comunicarse usando extensiones.
- trust-ad indica que el sistema confía en la información de autenticación de dominio (AD, Authenticated Data) proporcionada por el servidor DNS.
  Esto puede mejorar la seguridad al asegurar que la

información de respuesta DNS está autenticada.

search .:
Esta línea indica el dominio de búsqueda que se utiliza para completar nombres de dominio parciales que no están completamente calificados.
En este caso, el punto (.) indica que se debe buscar en el dominio raíz.
Si intentas resolver un nombre de dominio sin un sufijo, el sistema intentará agregar el sufijo especificado aquí antes de consultar al servidor DNS.

Tipos de servidores

Servidores Raíz DNS
Estos están en la parte superior de la jerarquía DNS, gestionando los dominios de nivel superior y interviniendo solo si los servidores de nivel inferior no responden. La Corporación de Internet para Nombres y Números Asignados (ICANN) supervisa su operación, con un conteo global de 13.

Servidores de Nombres Autorizados
Estos servidores tienen la última palabra para las consultas en sus zonas designadas, ofreciendo respuestas definitivas. Si no pueden proporcionar una respuesta, la consulta se eleva a los servidores raíz.

Servidores de Nombres No Autorizados
Sin propiedad sobre las zonas DNS, estos servidores recopilan información de dominio a través de consultas a otros servidores.

Servidor DNS de Caché
Este tipo de servidor memoriza las respuestas a consultas anteriores durante un tiempo determinado para acelerar los tiempos de respuesta para solicitudes futuras, con la duración de la caché dictada por el servidor autorizado.

Servidor de Reenvío
Cumpliendo un papel sencillo, los servidores de reenvío simplemente retransmiten consultas a otro servidor.

Resolutor
: Integrados en computadoras o enrutadores, los resolutores ejecutan la resolución de nombres localmente y no se consideran autorizados.

Host ubuntu resuelve ip

Orden común para obtener la ip de un nombre

/etc/hosts
: El sistema operativo primero verifica el archivo /etc/hosts para ver si hay una entrada correspondiente al nombre de dominio solicitado. Si existe una entrada en este archivo, se utiliza directamente y no se realiza ninguna consulta adicional.

Caché de resolución DNS
: Si la información no se encuentra en el archivo /etc/hosts o si la entrada ha expirado en la caché, el sistema operativo consulta su caché de resolución DNS local. La caché almacena las respuestas de consultas DNS anteriores y sus tiempos de vida (TTL). Si la información está en la caché y aún es válida, se utiliza directamente.

Consulta a servidores DNS configurados
: Si la información no está en la caché o ha expirado, el sistema operativo envía una solicitud DNS a los servidores DNS configurados en el archivo `/etc/resolv.conf`. Estos servidores pueden ser servidores DNS locales o servidores DNS públicos, según la configuración de red del sistema.

Resolución recursiva
: Si los servidores DNS configurados no tienen la información, realizarán consultas recursivas en nombre del host. Esto implica buscar la información en la jerarquía de servidores DNS hasta encontrar una respuesta autoritativa.

Actualización de la caché
: Una vez que se obtiene la respuesta del servidor DNS, se almacena en la caché local del sistema operativo para futuras consultas. La caché también se actualiza con nuevos registros DNS a medida que el sistema realiza más

consultas.

DNS Records

A (Address)
Proporciona la dirección IP correspondiente a un nombre de dominio.
Ejemplo: example.com A 192.168.1.1

AAAA (IPv6 Address)
Similar al registro A, pero para direcciones IPv6.
Ejemplo: example.com AAAA 2001:0db8:85a3:0000:0000:8a2e:0370:7334

CNAME (Canonical Name)
Asocia un alias (nombre de dominio) con un nombre de dominio real (canónico).
Ejemplo: www.example.com CNAME example.com

MX (Mail Exchange)
Especifica los servidores de correo responsables de recibir mensajes de correo electrónico para el dominio.
Ejemplo: example.com MX mail.example.com

TXT (Text)
Almacena texto arbitrario asociado con un nombre de dominio. Se usa para diversas finalidades, como verificación de la propiedad del dominio o configuración de políticas de seguridad.
Ejemplo: example.com TXT "Registro TXT de ejemplo"

PTR (Pointer)
Se utiliza para realizar búsquedas inversas y mapear una dirección IP a un nombre de dominio.
Ejemplo: 1.2.3.4 PTR host.example.com

NS (Name Server)
Indica los servidores de nombres autorizados para el dominio.
Ejemplo: example.com NS ns1.example.com

SOA (Start of Authority)
Define información sobre la administración del dominio, como el contacto del administrador, la fecha de la última actualización, etc.

Ejemplo: example.com SOA ns1.example.com admin.example.com 2023112001 7200 3600 1209600 86400

SRV (Service)

Proporciona información sobre servicios disponibles en la red, como servidores de correo, servidores de chat, etc.
Ejemplo: _sip._tcp.example.com SRV 10 60 5060 sipserver.example.com

CAA (Certification Authority Authorization)

Permite a un titular de dominio especificar qué entidades de certificación (CAs) están autorizadas a emitir certificados para ese dominio.
Ejemplo: example.com CAA 0 issue "letsencrypt.org"

DNS servers

Principales:

- **BIND9 (Berkeley Internet Name Domain, versión 9)** Es uno de los servidores DNS más tradicionales y robustos, diseñado para manejar entornos grandes y complejos.

  Características principales:

  - Flexibilidad: Admite múltiples funciones como DNS autoritativo, recursivo y soporte para DNSSEC.
  - Escalabilidad: Ideal para grandes infraestructuras y entornos empresariales.
  - Amplia documentación: Muy utilizado históricamente, por lo que cuenta con gran cantidad de recursos y una comunidad activa.
  - Configuración avanzada: Permite ajustes muy específicos, pero esto lo hace algo complejo para principiantes.

  Es para Empresas grandes con necesidades de DNS avanzadas y Servidores públicos que requieren alta confiabilidad y soporte para DNSSEC.

- **PowerDNS** (Servidor DNS moderno y modular) Diseñado para ser modular, moderno y altamente personalizable, PowerDNS destaca en entornos que requieren una integración estrecha con bases de datos.

  Características principales:

  - Modularidad: Puede funcionar como DNS autoritativo y resolver consultas de bases de datos SQL (MySQL, PostgreSQL) u otros backends como Redis.
  - Rendimiento: Optimizado para manejar altos

volúmenes de tráfico.
    - Integración con bases de datos: Ideal para entornos donde las zonas DNS están almacenadas y gestionadas en bases de datos.
    - DNSSEC: Soporte integrado con configuraciones más fáciles que BIND.
    - Interfaz moderna: Varios paneles de control de terceros lo soportan.

  Es para Proveedores de servicios de internet (ISPs) y grandes empresas que gestionan zonas DNS dinámicamente y para infraestructuras modernas que requieren flexibilidad e integración con otras herramientas.

- **dnsmasq** (Ligero y versátil para redes pequeñas). Un servidor DNS y DHCP ligero diseñado principalmente para redes pequeñas y medianas.

  Características principales:

    - Simplicidad: Fácil de configurar, ideal para entornos domésticos o pequeñas oficinas.
    - DNS en caché: Acelera la resolución de nombres al almacenar consultas previas.
    - Servidor DHCP integrado: Gestión sencilla de direcciones IP para dispositivos de la red.
    - Requerimientos bajos: Consume pocos recursos del sistema.

  Uso típicos son las Redes domésticas o pequeñas oficinas, dispositivos integrados o entornos con hardware limitado (por ejemplo, routers) y es ideal para configuraciones rápidas y ligeras que no necesitan funciones avanzadas como DNSSEC.

Comparativa rápida

Características	BIND9	PowerDNS	dnsmasq
Complejidad	Alta	Media	Baja
Escalabilidad	Muy alta	Muy alta	Baja
Facilidad de	Complejo	Moderado	Muy sencillo

configuración			
Consumo de recursos	Alto	Medio	Bajo
Soporte para DNSSEC	Sí	Sí	Limitado (no completo)
Backend basado en BD	No	Sí	No
Ideal para	Grandes empresas, ISP	ISP, infraestructuras modernas	Redes pequeñas

Otros a tener en cuenta

- **Unbound**. Unbound es un servidor DNS recursivo, seguro y ligero. Está diseñado para ser simple de configurar pero potente en términos de rendimiento y seguridad.
- **Knot DNS**. Knot DNS es un servidor autoritativo de alto rendimiento, desarrollado por CZ.NIC, el operador del dominio .CZ.
- **CoreDNS**. CoreDNS es un servidor DNS escrito en Go, diseñado para ser modular y extensible. Es particularmente popular en entornos de contenedores como Kubernetes.
- **djbdns**. Un servidor DNS creado por Daniel J. Bernstein, conocido por su diseño seguro y de alto rendimiento.
- **NSD (Name Server Daemon)**. NSD es un servidor DNS autoritativo simple y eficiente, desarrollado para operar grandes zonas de forma segura y con alto rendimiento.
- **MaraDNS**. MaraDNS es un servidor DNS enfocado en la simplicidad y seguridad, especialmente para redes pequeñas y medianas.
- **Pi-hole**. Pi-hole es una herramienta que actúa como un servidor DNS y bloqueador de publicidad para redes locales.
- **Cloudflare's 1.1.1.1.** Aunque no es un servidor que los alumnos configuren directamente, Cloudflare proporciona un servicio DNS rápido y con funcionalidades avanzadas, como DNS sobre HTTPS (DoH).

Host

- Consulta de una dirección IP para obtener un nombre de dominio:

  ```
  host 8.8.8.8
  ```

  Esto buscará el nombre de dominio asociado con la dirección IP 8.8.8.8 utilizando DNS.

- Consulta de un nombre de dominio para obtener direcciones IP:

  ```
  host www.example.com
  ```

  Esto buscará las direcciones IP asociadas con el nombre de dominio www.example.com.

- Consulta inversa para obtener un nombre de dominio desde una dirección IP:

  ```
  host 216.58.205.46
  ```

  Esto realizará una consulta inversa (reverse DNS lookup) para encontrar el nombre de dominio asociado con la dirección IP 216.58.205.46.

- Consulta de registros DNS específicos:

  ```
  host -t MX example.com
  ```

  Esto buscará los registros de intercambio de correo (MX records) asociados con el dominio example.com.

- Consulta de registros DNS de tipo AAAA (IPv6):

```
host -t AAAA www.example.com
```

Esto buscará las direcciones IPv6 asociadas con el nombre de dominio www.example.com.

- Consulta de registros DNS de tipo SOA (Start of Authority):

```
host -t SOA example.com
```

Esto mostrará la información de autoridad de inicio (Start of Authority) para el dominio example.com.

- Consulta de servidores de nombres para un dominio:

```
host -t NS example.com
```

Esto mostrará los servidores de nombres autoritativos para el dominio example.com.

Whois

- Consulta de información sobre un nombre de dominio:

```
whois example.com
```

Esto proporcionará información sobre el propietario registrado del dominio example.com, la fecha de registro y otra información relacionada con el dominio.

- Consulta de información sobre una dirección IP:

```
whois 8.8.8.8
```

Esto mostrará información relacionada con la dirección IP 8.8.8.8, como el rango de IP, el propietario registrado y otros detalles asociados.

De los datos que se obtienen podemos resaltar:

- **Para el rango 8.0.0.0 - 8.127.255.255**

  NetRange
  Rango de direcciones IP asignado. En este caso, desde 8.0.0.0 hasta 8.127.255.255.

  CIDR
  Notación de prefijo CIDR asociada con el rango de direcciones IP. En este caso, 8.0.0.0/9.

  NetName
  Nombre asignado a este rango de direcciones IP. En este caso, "LVLT-ORG-8-8".
  Es una etiqueta que ayuda a identificar y etiquetar la asignación de direcciones IP.

  NetHandle
  Identificador único asignado a este rango de direcciones IP. En este caso, "NET-8-0-0-0-1".
  Es utilizado para distinguir de manera única una asignación de direcciones IP en las bases de datos WHOIS

  Parent
  Indica el rango padre al que pertenece este rango. En este caso, "NET8 (NET-8-0-0-0-0)".

  NetType

  Tipo de asignación directa. En este caso, "Direct Allocation".

  Direct Allocation (Asignación Directa)
  Las direcciones IP en este rango han sido asignadas directamente a una entidad (como una empresa o proveedor de servicios) por una organización de asignación de direcciones IP, como ARIN (American Registry for Internet Numbers) en el

caso de direcciones IPv4 en América del Norte.
Ejemplo: NetType: Direct Allocation

Allocated to Hosting Company (Asignado a una Empresa de Alojamiento)
Este tipo indica que el bloque de direcciones IP ha sido asignado a una empresa de alojamiento que proporciona servicios de alojamiento web.
Ejemplo: NetType: Allocated to Hosting Company

Allocated to ISP (Asignado a un Proveedor de Servicios de Internet)
Este tipo indica que el bloque de direcciones IP ha sido asignado a un proveedor de servicios de Internet (ISP) para su distribución a sus clientes.
Ejemplo: NetType: Allocated to ISP

Allocated to End User (Asignado a un Usuario Final)
Las direcciones IP han sido asignadas directamente a un usuario final o cliente. Esto podría ser una empresa o una entidad que utiliza las direcciones IP para sus propios fines.
Ejemplo: NetType: Allocated to End User

OriginAS
Identificador de sistema autónomo de origen (AS). No se proporciona en este caso.

Organization
Nombre de la organización propietaria de este rango. En este caso, "Level 3 Parent, LLC (LPL-141)".

RegDate
Fecha de registro inicial. En este caso, "1992-12-01".

Updated
Fecha de última actualización de la información. En este caso, "2018-04-23".

Luego sigue la información detallada sobre la organización propietaria, incluyendo la dirección, ciudad, estado, código postal y país. También se proporcionan comentarios relevantes sobre el uso de este espacio de direcciones IP, términos de servicio, y cómo abordar problemas de abuso.

- **Para el rango específico 8.8.8.0 - 8.8.8.255**
  - La información es similar a la anterior, pero ahora está específicamente relacionada con el rango 8.8.8.0/24.
  - La organización propietaria es "Google LLC" (GOGL).
  - Se proporciona información de contacto para asuntos de abuso y técnicos.

- Consulta de información sobre un rango de direcciones IP (CIDR):

```
whois 192.168.0.0/24
```

Esto proporcionará información sobre el bloque de direcciones IP especificado, incluido el propietario registrado y detalles relacionados.

- Consulta de información sobre un bloque de direcciones IP asignado a un país:

```
whois -h whois.radb.net -- -i origin -T route
8.8.8.0/24
```

Esto buscará información sobre el bloque de direcciones IP 8.8.8.0/24 utilizando la base de datos RADB (Routing Assets Database). Ten en cuenta que la opción -h se utiliza para especificar un servidor WHOIS específico.

> ***Nota: Enlace con Seguridad (OSINT)***
>
> *Desde la perspectiva de la seguridad informática, el comando whois es una herramienta esencial de OSINT (Open Source Intelligence).*
> *La información que proporciona sobre el propietario de un dominio, las fechas de registro y, especialmente, los rangos de direcciones IP (NetRange) y la organización responsable (Organization), constituye a menudo el primer paso crítico en la fase de reconocimiento de un ataque informático, denominada Footprinting.*
> *Al analizar datos como el identificador del sistema autónomo o el tipo de asignación de red (NetType), un atacante puede determinar si la infraestructura es propia, está alojada en un proveedor de servicios (ISP) o pertenece a una empresa de hosting. Estos detalles permiten trazar un mapa de la superficie de exposición de la organización, identificar posibles objetivos para ataques de ingeniería social mediante los contactos técnicos y de abuso, o definir los segmentos de red específicos sobre los que se realizarán escaneos de vulnerabilidades posteriores.*

Ver también el artículo: traceroute y mtr

resolvectl

Proporciona información sobre la configuración actual de resolución de nombres en tu sistema, incluyendo detalles sobre las interfaces de red y los servidores DNS utilizados.

```
resolvectl
Global
       Protocols: -LLMNR -mDNS -DNSOverTLS DNSSEC=no/
unsupported
resolv.conf mode: stub
```

```
Link 2 (eno1)
Current Scopes: none
     Protocols: -DefaultRoute +LLMNR -mDNS -DNSOverTLS
DNSSEC=no/unsupported

Link 3 (wlp1s0)
    Current Scopes: DNS
         Protocols: +DefaultRoute +LLMNR -mDNS -
DNSOverTLS DNSSEC=no/unsupported
Current DNS Server: 80.58.61.250
       DNS Servers: 80.58.61.250 80.58.61.254

Link 4 (docker0)
Current Scopes: none
     Protocols: -DefaultRoute +LLMNR -mDNS -DNSOverTLS
DNSSEC=no/unsupported
```

- **Sección Global**

  Protocols

  Indica los protocolos de resolución de nombres activos o desactivados a nivel global en el sistema.

  - se desactivaron LLMNR (Link-Local Multicast Name Resolution) y mDNS (Multicast DNS), y
  - DNSOverTLS no es compatible o no está habilitado.
  - DNSSEC (DNS Security Extensions) también está desactivado o no es compatible.

  resolv.conf mode: stub

  Indica que el modo de resolv.conf en el sistema es "stub". En este modo, systemd-resolved puede generar dinámicamente el archivo resolv.conf según la configuración actual.

  En términos generales, hay tres modos principales para el archivo resolv.conf gestionado por systemd-resolved:

1. **resolv.conf mode: default:** En este modo, systemd-resolved no gestiona el archivo resolv.conf, y es responsabilidad del sistema o del administrador del sistema mantener y actualizar manualmente el archivo. systemd-resolved no sobrescribe el contenido de resolv.conf.
2. **resolv.conf mode: static:** En este modo, systemd-resolved administra el archivo resolv.conf, pero no lo actualiza dinámicamente.
   Es útil si deseas gestionar manualmente el contenido del archivo resolv.conf and no deseas que systemd-resolved lo modifique.
3. **resolv.conf mode: stub:** En este modo, systemd-resolved genera dinámicamente el archivo resolv.conf basándose en la información de resolución de nombres disponible en el sistema.
   Este modo es útil cuando deseas que systemd-resolved maneje la configuración del archivo resolv.conf de manera automática.

La configuración específica de stub indica que systemd-resolved actúa como un "modificador de resolv.conf" en lugar de gestionar directamente la resolución DNS.

- Si deseas activar el modo stub (predeterminado):

```
DNSStubListener=yes
```

- Si deseas desactivar el modo stub y permitir que systemd-resolved actúe como un servidor DNS local completo:

```
DNSStubListener=no
```

- **Sección de Link 2 (eno1)**

  Current Scopes: none
  Indica que la interfaz de red "eno1" no tiene ámbitos de resolución de nombres activos actualmente.
  Protocols: -DefaultRoute +LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
  Muestra los protocolos de resolución de nombres activos o desactivados para la interfaz "eno1".
  En este caso, se desactivó el protocolo por defecto de ruta (-DefaultRoute), se activó LLMNR, y los demás protocolos están en el mismo estado que a nivel global.

- **Sección de Link 3 (wlp1s0)**

  Current Scopes: DNS
  Indica que la interfaz de red "wlp1s0" tiene un ámbito de resolución de nombres activo y está configurada para usar servidores DNS.
  Protocols: +DefaultRoute +LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
  Muestra los protocolos de resolución de nombres activos o desactivados para la interfaz "wlp1s0".
  En este caso, se activó el protocolo por defecto de ruta (+DefaultRoute), se activó LLMNR, y los demás protocolos están en el mismo estado que a nivel global.
  IMPORTANTE: wlp1s0 utiliza el "protocolo por defecto de ruta" para las consultas de resolución de nombres.
  En este caso, +DefaultRoute significa que este protocolo está habilitado.
  La importancia de esto radica en que, al tener habilitado el "protocolo por defecto de ruta," las consultas de resolución de nombres seguirán las configuraciones globales o predeterminadas del sistema cuando no haya una configuración específica para la interfaz en cuestión.
  Este protocolo es utilizado cuando el sistema necesita

resolver nombres de dominio y no hay configuraciones más específicas para la interfaz que maneja la comunicación de red.

Current DNS Server: 80.58.61.250
Indica el servidor DNS actualmente configurado para la interfaz "wlp1s0".

DNS Servers: 80.58.61.250 80.58.61.254
Muestra la lista de servidores DNS configurados para la interfaz "wlp1s0".

- **Sección de Link 4 (docker0)**

Current Scopes: none
Indica que la interfaz de red "docker0" no tiene ámbitos de resolución de nombres activos actualmente.

Protocols: -DefaultRoute +LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
Muestra los protocolos de resolución de nombres activos o desactivados para la interfaz "docker0". En este caso, se desactivó el protocolo por defecto de ruta (-DefaultRoute), se activó LLMNR, y los demás protocolos están en el mismo estado que a nivel global.

nslookup

Respuesta Autoritativa (Authoritative Answer):
Una respuesta autoritativa proviene directamente del servidor DNS que es la fuente de autoridad para el dominio consultado.
Este servidor es responsable de mantener la información actualizada para ese dominio.
La respuesta autoritativa es la fuente más confiable de información.

Respuesta No Autoritativa (Non-authoritative Answer):
Una respuesta no autoritativa significa que la información proviene de un servidor DNS que no es la fuente de

autoridad principal para el dominio consultado.
Puede provenir de una caché local o de un servidor intermedio que ha almacenado previamente la información.

Ejemplo:

```
$ nslookup www.google.es
Server:  127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: www.google.es
Address: 142.251.167.94
Name: www.google.es
Address: 2a00:1450:4003:80f::2003
```

Server y Address:
Indican la dirección IP del servidor DNS al que se realizó la consulta (127.0.0.53 en este caso).
Esta dirección IP es local (127.0.0.53), lo que sugiere que la consulta se realizó a un servidor DNS en la máquina local.
(Nota: 127.0.0.53 es la IP del stub resolver de systemd-resolved y no necesariamente el servidor DNS final)

Non-authoritative answer:
La respuesta es no autoritativa.
La información no proviene directamente de un servidor DNS autoritativo para el dominio "www.google.es".
En cambio, la respuesta se obtuvo de una fuente no autoritativa, que podría ser una caché local o un servidor DNS intermedio.

Name y Address:
La dirección IP asociada con "www.google.es" es 142.251.167.94 (IPv4) y 2a00:1450:4003:80f::2003 (IPv6).

Ejemplo DNS inverso:

```
nslookup 8.8.8.8
8.8.8.8.in-addr.arpa name = dns.google.
```

"in-addr.arpa" es una convención utilizada en el DNS para la resolución inversa de direcciones IP, permitiendo asociar direcciones IP con nombres de dominio.

nameserver 127.0.0.53
: Esta línea especifica la dirección IP del servidor DNS al que el sistema enviará consultas de resolución de nombres. En este caso, se está configurando el sistema para utilizar un servidor DNS local en la dirección IP 127.0.0.53. Esta dirección IP es comúnmente utilizada por systemd-resolved, un sistema de resolución de nombres que puede estar presente en sistemas Linux modernos.

options edns0 trust-ad
: Esta línea especifica algunas opciones de configuración para las consultas DNS. edns0 se refiere a la extensión de DNS (EDNS) que permite a los clientes y servidores DNS comunicarse usando extensiones. trust-ad indica que el sistema confía en la información de autenticación de dominio (AD, Authenticated Data) proporcionada por el servidor DNS. Esto puede mejorar la seguridad al asegurar que la información de respuesta DNS está autenticada.

search .
: Esta línea indica el dominio de búsqueda que se utiliza para completar nombres de dominio parciales que no están completamente calificados. En este caso, el punto (.) indica que se debe buscar en el dominio raíz. Si intentas resolver un nombre de dominio sin un sufijo, el sistema intentará agregar el sufijo especificado aquí antes de consultar al servidor DNS.

Principales opciones:

- Respuesta desde la Caché Local:

  Si la consulta ya se ha realizado recientemente y la respuesta está en la caché local, nslookup puede devolver la información almacenada en la caché, que no necesariamente proviene del servidor DNS autoritativo.

  ```
  nslookup example.com
  ```

- Respuesta desde un Servidor de Reenvío:

  Si tu sistema está configurado para utilizar un servidor de reenvío (forwarder), nslookup puede obtener la respuesta de ese servidor en lugar de realizar la búsqueda directamente al servidor DNS autoritativo.

  ```
  nslookup example.com 8.8.8.8
  ```

- Consulta Directa al Servidor Autoritativo:

  Puedes forzar a nslookup para que realice la consulta directamente al servidor DNS autoritativo usando la opción -norecurse.

  ```
  nslookup –norecurse example.com
  ```

Dig

Heredero de nslookup

```
sudo apt install dnsutils
```

- Consulta de registros de tipo A (IPv4):

  ```
  dig example.com
  ```

  Esto mostrará las direcciones IPv4 asociadas con el nombre de dominio example.com.
  Ejemplo:

  ```
  dig www.google.es
  ```

```
; <<>> DiG 9.18.18-0ubuntu0.23.04.1-Ubuntu <<>>
www.google.es
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
30360
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY:
0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.google.es.   IN A

;; ANSWER SECTION:
www.google.es.  9 IN A 142.250.200.131

;; Query time: 184 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Tue Nov 21 12:38:31 CET 2023
;; MSG SIZE  rcvd: 58
```

Encabezado (HEADER)

opcode: QUERY
Indica que la operación es una consulta.
status: NOERROR
Indica que la consulta se completó con éxito.
id: 30360
Es un identificador único para la consulta.
flags: qr rd ra

qr Indica que es una respuesta (1 para respuestas, 0 para consultas).

qr: 1
Respuesta a una Consulta: Indica que la respuesta es una respuesta a una consulta.
Este es el caso más común y significa que la respuesta contiene la información

solicitada en la consulta DNS.

qr: 0

Consulta (No es una Respuesta):
Indica que el mensaje DNS es una consulta y no una respuesta.
Cuando qr: 0, la respuesta completa a la consulta aún no se ha incluido en el mensaje actual; está pendiente de recibir desde el servidor DNS.
El cliente (quien realiza la consulta) está solicitando al servidor DNS que proporcione la información correspondiente al nombre de dominio o recurso solicitado.

rd - Indica que se solicitó recursividad

rd: 1

indica que el cliente (quien realiza la consulta) solicitó recursividad.
El cliente está pidiendo al servidor de nombres que, en caso de no tener la respuesta en su caché, realice consultas adicionales a otros servidores de nombres para obtener la respuesta completa.

rd: 0

indicaría que no se solicitó recursividad.

ra - Indica que el servidor de nombres admite la recursividad.

ra: 1

indica que el servidor de nombres que respondió admite la recursividad.
El servidor de nombres tiene la capacidad de realizar consultas a otros servidores de nombres en caso de necesitar obtener información adicional para responder a la consulta.

ra: 0

indicaría que el servidor de nombres que respondió no admite la recursividad y solo responde con la información que tiene en su caché.

ADDITIONAL

proporciona información adicional relevante para la respuesta principal obtenida en la sección "ANSWER".
La sección "ADDITIONAL" generalmente incluye registros DNS que son relevantes para el procesamiento de la respuesta principal.

Registros de Autoridad (NS)

Estos son registros que indican los servidores de nombres autoritarios para la zona consultada.
Proporcionan información sobre qué servidores de nombres son responsables de la zona.

Registros de Recursos (A, AAAA, CNAME, etc.)

Pueden incluir registros de recursos

adicionales relacionados con la respuesta principal, como direcciones IP (A o AAAA) o alias de nombres (CNAME).

Registros de Soporte de Extensiones de DNS (EDNS)

Pueden incluir información adicional relacionada con la negociación de extensiones de DNS, como el tamaño máximo de paquete que el servidor puede manejar.

Nota: En muchas ocasiones aunque tenga ADITIONAL: 1 se obvia y no se muestra una sección al respecto.

Sección de Pseudosección OPT

EDNS

Indica que se está utilizando Extension Mechanisms for DNS (EDNS).
Extension Mechanisms for DNS (EDNS) es una extensión del protocolo DNS (Domain Name System) que permite la inclusión de información adicional y funcionalidades más avanzadas en las consultas y respuestas DNS. La inclusión de EDNS se refleja en la sección de "OPT PSEUDOSECTION" en la salida del comando dig.

Tamaño de Mensaje Aumentado

EDNS permite el uso de mensajes DNS más grandes que el límite tradicional de 512 bytes de UDP. Esto es útil para consultas y respuestas DNS que contienen información adicional.

Soporte para IPv6

EDNS facilita la transición a IPv6 al permitir la inclusión de registros

IPv6 y la comunicación a través de conexiones IPv6.

Adición de Opciones

EDNS permite la inclusión de opciones adicionales en las consultas y respuestas DNS.
Esto puede incluir información como el soporte para DNSSEC (DNS Security Extensions), la capacidad de fragmentar respuestas DNS grandes, y otros aspectos que pueden mejorar la funcionalidad y seguridad del sistema DNS.

Negociación de Características

EDNS facilita la negociación de características entre el cliente y el servidor DNS, permitiendo que ambas partes acuerden el uso de extensiones específicas.

version: 0

La versión de EDNS es 0.

udp: 65494

Indica el tamaño máximo del mensaje UDP.

Sección de Preguntas (QUESTION SECTION)

www.google.es. IN A: La consulta realizada fue para obtener el registro de tipo A (dirección IPv4) del nombre de dominio www.google.es.

Sección de Respuestas (ANSWER SECTION)

www.google.es. 9 IN A 142.250.200.131: La respuesta indica que la dirección IP asociada con www.google.es es 142.250.200.131. El tiempo de vida (TTL) es de 9 segundos.

Tiempo de Consulta (Query time)

184 msec: Indica el tiempo que tardó la consulta en ser respondida.

Servidor de Nombres (SERVER)

127.0.0.53#53(127.0.0.53) (UDP): Indica que la consulta se envió al servidor de nombres local en el

puerto 53 utilizando UDP.

Marca de Tiempo (WHEN)
Tue Nov 21 12:38:31 CET 2023: Muestra la fecha y la hora en que se recibió la respuesta.

Tamaño del Mensaje Recibido (MSG SIZE rcvd)
58: Indica el tamaño del mensaje recibido en bytes.

- Consulta de registros de tipo AAAA (IPv6):

```
dig AAAA www.example.com
```

Esto mostrará las direcciones IPv6 asociadas con el nombre de dominio www.example.com.

- Consulta inversa para obtener un nombre de dominio desde una dirección IP:

```
dig -x 216.58.205.46
```

Esto realizará una consulta inversa (reverse DNS lookup) para encontrar el nombre de dominio asociado con la dirección IP 216.58.205.46.

```
dig -x 216.58.205.46

; <<>> DiG 9.18.18-0ubuntu0.23.04.1-Ubuntu <<>> -x
216.58.205.46
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
10333
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY:
0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;46.205.58.216.in-addr.arpa. IN PTR

;; ANSWER SECTION:
46.205.58.216.in-addr.arpa. 11815 IN PTR lhr48s23-
in-f14.1e100.net.
```

```
46.205.58.216.in-addr.arpa. 11815 IN PTR mil07s19-
in-f14.1e100.net.
46.205.58.216.in-addr.arpa. 11815 IN PTR mil04s24-
in-f46.1e100.net.
46.205.58.216.in-addr.arpa. 11815 IN PTR mil04s24-
in-f14.1e100.net.

;; Query time: 27 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Fri Nov 24 14:01:46 CET 2023
;; MSG SIZE  rcvd: 184
```

46.205.58.216.in-addr.arpa.
: Este es el formato del nombre de dominio invertido asociado con la dirección IP 216.58.205.46. En una búsqueda inversa de DNS, las direcciones IP se representan en el dominio in-addr.arpa., y las partes de la dirección se invierten y se colocan como subdominios.

11815 IN PTR
: Esta es la información de tiempo de vida (TTL) y el tipo de registro. En este caso, el TTL es 11815 segundos, e IN indica un registro de Internet.

lhr48s23-in-f14.1e100.net., mil07s19-in-f14.1e100.net., mil04s24-in-f46.1e100.net., mil04s24-in-f14.1e100.net.:
: Estos son los nombres de host asociados con la dirección IP 216.58.205.46. La dirección IP puede tener varios nombres de host asociados, y estos nombres de host generalmente están vinculados a diferentes ubicaciones o servidores de Google. Cada nombre de host está asociado con un servidor específico de la infraestructura de Google (1e100.net es un dominio utilizado por Google para varios propósitos).

- Consulta de registros de tipo MX (Mail Exchange):

```
dig MX example.com
```

Esto mostrará los registros de intercambio de correo (MX records) asociados con el dominio example.com.

- Consulta de registros de tipo TXT (Text):

```
dig TXT example.com
```

Esto mostrará los registros de texto asociados con el dominio example.com, que a menudo se utilizan para almacenar información de texto arbitraria.

- Consulta de servidores de nombres autoritativos para un dominio:

```
dig NS example.com
```

Esto mostrará los servidores de nombres autoritativos para el dominio example.com.

- Consulta de registros de tipo SOA (Start of Authority):

```
dig SOA example.com
```

Esto mostrará la información de autoridad de inicio (Start of Authority) para el dominio example.com.
Ejemplo dig SOA www.google.es

```
dig SOA www.google.es
; <<>> DiG 9.18.18-0ubuntu0.23.04.1-Ubuntu <<>> SOA
www.google.es
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
22495
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY:
1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.google.es.   IN SOA

;; AUTHORITY SECTION:
google.es.  60 IN SOA ns1.google.com. dns-
```

```
admin.google.com. 585242313 900 900 1800 60

;; Query time: 47 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Sun Nov 26 11:18:18 CET 2023
;; MSG SIZE  rcvd: 102
```

- QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1:
  Se realizó una consulta (QUERY), pero no se obtuvo una respuesta directa (ANSWER: 0).
  Sin embargo, se proporciona información de autoridad (AUTHORITY: 1) y se incluye información adicional (ADDITIONAL: 1).
  Nota: si obtienes AUTHORITY: 2, significa que hay 2 registros de autoridad en la respuesta DNS para la consulta. Estos registros de autoridad son parte de la información devuelta por el servidor DNS y ayudan a identificar quién es la autoridad para la zona del dominio consultado.
- Sección de Pregunta (QUESTION SECTION):
  www.google.es. IN SOA: Muestra la sección de pregunta indicando que se consultó el registro SOA para www.google.es.
- google.es. 60 IN SOA ns1.google.com. dns-admin.google.com. 585242313 900 900 1800 60
  Proporciona información de autoridad para el dominio google.es.
  Indica que el servidor de nombres autoritativo (ns1.google.com.) es responsable del dominio.
  También incluye información sobre la zona, como el número de serie y los intervalos de renovación, reintentos y expiración.

Número de Serie (Serial Number)

El número de serie es un valor que se incrementa cada vez que se realiza una modificación en la zona. Su propósito principal es permitir que otros servidores DNS determinen si la información de su copia de la zona está actualizada.
Cuando se realiza una actualización en la zona (por

ejemplo, al agregar o modificar registros), el número de serie se incrementa. Los demás servidores DNS que tienen una copia de la zona pueden comparar el número de serie para determinar si necesitan actualizar su copia.

Intervalo de Renovación (Refresh Interval)
El intervalo de renovación especifica con qué frecuencia un servidor secundario debe intentar obtener una copia actualizada de la zona desde el servidor maestro (autoritativo).
Si un servidor secundario no puede comunicarse con el servidor maestro dentro de este intervalo, intentará renovar la zona después de este tiempo.

Intervalo de Reintentos (Retry Interval)
El intervalo de reintentos indica cuánto tiempo debe esperar un servidor secundario antes de volver a intentar obtener una copia de la zona después de un intento fallido durante el intervalo de renovación.
Si un servidor secundario no puede renovar la zona con éxito durante el intervalo de renovación, esperará el intervalo de reintentos antes de hacer otro intento.

Intervalo de Expiración (Expire Interval)
El intervalo de expiración es el tiempo que un servidor secundario espera antes de considerar que su copia de la zona está obsoleta si no puede renovarla con éxito durante el intervalo de renovación y el intervalo de reintentos.
Después de que ha pasado el intervalo de expiración sin obtener una copia actualizada, el servidor secundario deja de responder a consultas para la zona y considera que su copia de la zona ha expirado.

- Si queremos consultar a un dns en particular lo hacemos de la siguiente manera:

```
dig @8.8.8.8 example.com
```

`systemd-resolved` es un componente del sistema de inicio `systemd` (Sistema de inicialización y gestión de servicios para sistemas operativos Linux).
`systemd-resolved` proporciona servicios de resolución de nombres y gestión de DNS en sistemas que lo utilizan.

Funciones y servicios que ofrece `systemd-resolved`:

Resolución de Nombres Local:
: Mantiene una base de datos local de nombres de host y direcciones IP para mejorar la velocidad de resolución y reducir la dependencia de consultas externas al servidor DNS.

Servidor DNS Local:
: Actúa como un servidor DNS local en la dirección IP 127.0.0.53 por defecto. Acepta consultas DNS de aplicaciones en el sistema y las resuelve utilizando información local o consultando servidores DNS remotos según sea necesario.
(Nota: 127.0.0.53 es la IP del stub resolver de systemd-resolved y no necesariamente el servidor DNS final.)

Soporte para DNS sobre TLS (DoT) y DNS sobre HTTPS (DoH):
: Puede ser configurado para utilizar protocolos seguros como DNS sobre TLS y DNS sobre HTTPS para cifrar las consultas y respuestas DNS, mejorando la privacidad y seguridad de las comunicaciones DNS.

Resolución Multicast DNS (mDNS):

: Puede realizar la resolución de nombres mediante el protocolo mDNS, que permite la resolución de nombres en redes locales sin la necesidad de un servidor DNS central.
Ver: mDNS (Multicast DNS)

: Se puede deshabilitar del sistema de una de las siguientes formas:

- Deshabilitar el servicio Avahi (que proporciona implementación mDNS):

```
sudo systemctl stop avahi-daemon
sudo systemctl disable avahi-daemon
```

- Configurar systemd-resolved para Ignorar mDNS en `/etc/systemd/resolved.conf`:

```
MulticastDNS=no
```

Implementación de Extensiones de DNS (EDNS):
Soporta extensiones de DNS para permitir la comunicación de información adicional entre el cliente y el servidor DNS, como la autenticación de datos (AD) que indica la autenticidad de la respuesta.

Cacheo de Resultados DNS:
Almacena en caché los resultados de las consultas DNS, lo que puede mejorar el rendimiento al evitar consultas repetitivas para los mismos nombres de dominio.

Cómo `systemd-resolved` maneja la resolución de nombres local y mantiene su caché

Caché de Resolución Local:
`systemd-resolved` mantiene una caché local en la memoria del sistema para almacenar las consultas DNS previas. Cuando se realiza una resolución de nombres, los resultados se almacenan en esta caché durante un tiempo determinado.

Tiempo de Vida (TTL):
Cada entrada en la caché tiene un Tiempo de Vida (TTL) asociado. El TTL especifica cuánto tiempo debe mantenerse la entrada en la caché antes de que expire. Si una aplicación vuelve a realizar la misma consulta antes de que

expire el TTL, `systemd-resolved` puede devolver la respuesta directamente desde la caché sin realizar una nueva consulta a través de la red.

Actualización Automática:

`systemd-resolved` puede actualizar automáticamente la caché según el tiempo transcurrido y el TTL asociado a cada entrada. Esto significa que las entradas obsoletas se eliminan de la caché después de que haya pasado su tiempo de vida.

Configuración del Tiempo de Vida:

Puedes ajustar la configuración del tiempo de vida en el archivo de configuración `/etc/systemd/resolved.conf`. Busca la configuración relacionada con "DNS" y "DNSStubListener". La configuración específica puede variar según la distribución, pero un ejemplo podría ser:

```
[Resolve]
DNS=...
# Otros parámetros
Cache=yes
CacheSize=64M
CacheLimit=32K
```

En esta configuración, CacheSize especifica el tamaño máximo de la caché, y CacheLimit especifica el número máximo de elementos en la caché. El tiempo de vida de las entradas de caché también se puede ajustar con otros parámetros relacionados con el DNS.

Diferencia entre Soporte para DNS sobre TLS (DoT) y DNS sobre HTTPS (DoH)

DNS sobre TLS (DoT) y DNS sobre HTTPS (DoH) son dos protocolos diseñados para mejorar la seguridad y la privacidad en las consultas DNS al cifrar la comunicación entre el cliente y el servidor DNS. Aunque ambos tienen el objetivo general de proporcionar una capa adicional de seguridad, hay diferencias

clave en cómo implementan esta protección y en sus enfoques técnicos.

DNS sobre TLS (DoT):

Transporte Seguro:
DoT utiliza el protocolo de seguridad TLS (Transport Layer Security) para cifrar las comunicaciones DNS entre el cliente y el servidor. Opera en el puerto 853.

Integración con el Sistema de Nombres de Dominio (DNS):
DoT se integra directamente con el sistema de nombres de dominio existente y utiliza la misma infraestructura, pero agrega una capa de seguridad adicional al utilizar TLS.

Puerto Específico:
DoT utiliza un puerto específico (853 por defecto) para el tráfico cifrado, lo que facilita su identificación y enrutamiento. Los puertos predeterminados para DNS son 53 (no cifrado), y cuando utilizas DoT (DNS sobre TLS), el puerto predeterminado es 853.

Configuración de systemd-resolved con DoT:

1. Edita el archivo de configuración resolved.conf:

```
sudo nano /etc/systemd/resolved.conf
```

2. Agrega las siguientes líneas para configurar DoT:

```
[Resolve]
DNS=IP_DEL_SERVIDOR#853
```

3. Reemplaza IP_DEL_SERVIDOR con la dirección IP del servidor DNS que admita DoT. Guarda los cambios y reinicia systemd-resolved:

```
sudo systemctl restart systemd-resolved
```

Esto indica a systemd-resolved que debe utilizar DoT en el puerto 853.

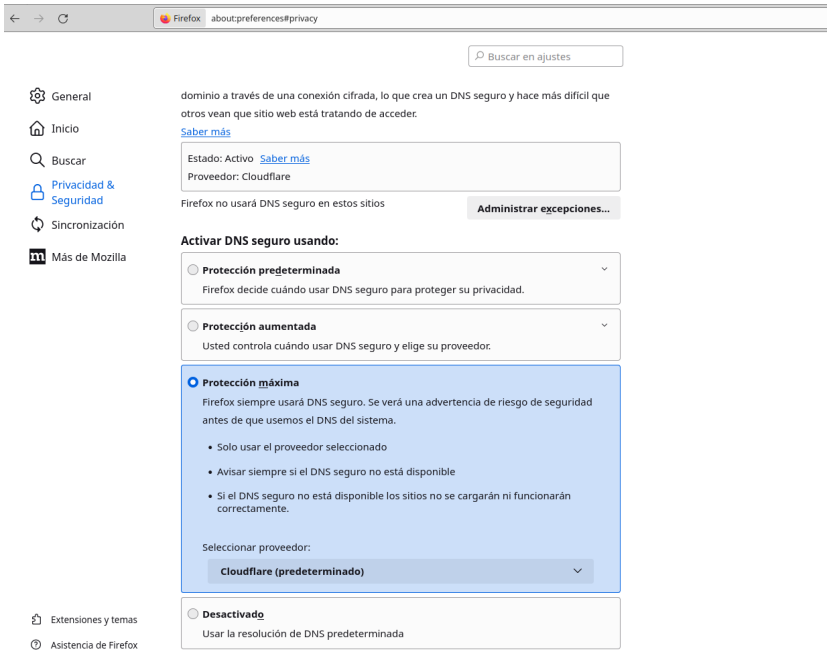
4. Para el bind9, si lo queremos usar en el forward haríamos en `/etc/bind/named.conf.options`:

```
options {
    directory "/var/cache/bind";

    // Otras configuraciones aquí...

    forwarders {
        // Aquí puedes especificar tus servidores
DNS forwarders con DoT
        1.1.1.1 port 853;  // Ejemplo con
Cloudflare DoT
        9.9.9.9 port 853;  // Ejemplo con Quad9 DoT
        // Agrega otros servidores DoT según sea
necesario
    };

    // Otras configuraciones aquí...
};
```

DNS sobre HTTPS (DoH):

DNS sobre HTTPS (DoH) is un protocolo que proporciona una capa adicional de seguridad a las consultas DNS al cifrarlas mediante HTTPS. Tradicionalmente, las consultas DNS se realizan en texto sin cifrar, lo que potencialmente permite que terceros intercepten o manipulen estas consultas. DoH resuelve este problema al encapsular las consultas DNS en conexiones HTTPS seguras.

write_enable



Cifrado a través de HTTPS:
DoH utiliza HTTPS para cifrar las consultas DNS. Opera en el puerto estándar 443, el mismo que se utiliza para HTTPS, lo que puede ayudar a evadir la detección y bloqueo del tráfico DNS.

Integración con Navegadores Web:
DoH se ha implementado inicialmente en navegadores web, lo que permite que las consultas DNS se realicen a través de la infraestructura HTTPS existente. Esto facilita su despliegue, especialmente en entornos controlados por proveedores de servicios de Internet (ISPs).

Mayor Despliegue en el Entorno de Navegadores:
DoH ha ganado popularidad en el entorno de navegadores web, donde se implementa como una option de configuración o se habilita automáticamente en algunos navegadores. Esto ha llevado a un despliegue más rápido en comparación con DoT.

Integración con Proveedores de Servicios de Internet (ISPs):
Al utilizar el puerto 443, DoH puede ser más difícil de bloquear o limitar, ya que se parece al tráfico HTTPS

regular. Esto ha llevado a debates sobre la neutralidad de la red y la capacidad de los ISPs para inspeccionar o gestionar el tráfico DNS.

Puerto Estándar:
El puerto estándar utilizado por DoH es el 443, que es el mismo puerto utilizado para HTTPS. Esto permite que las consultas DNS se mezclen con el tráfico HTTPS normal, facilitando la implementación en redes que puedan bloquear o limitar el tráfico DNS.

Privacidad:
DoH mejora la privacidad del usuario al ocultar las consultas DNS de los intermediarios no deseados, como proveedores de servicios de Internet (ISPs) o redes públicas.

Compatibilidad con Navegadores:
Muchos navegadores web modernos han comenzado a incorporar soporte nativo para DoH. Esto significa que, si está habilitado, el navegador puede realizar consultas DNS utilizando DoH sin requerir configuraciones adicionales.

Configuración Personalizada:
Además de la integración del navegador, también es posible configurar sistemas operativos o aplicaciones para utilizar servidores DNS que admitan DoH.

Proveedores de DNS que Ofrecen DoH:
Varios proveedores de DNS, como Cloudflare (1.1.1.1), Google (8.8.8.8), y otros, han implementado soporte para DoH. Cada proveedor puede tener su propia URL específica para DoH que se utiliza para enviar las consultas DNS cifradas.

Ejemplo para Bind9:

```
forwarders {
    https://1.1.1.1/dns-query;  # Cloudflare DNS sobre
HTTPS
};
```

o

```
forwarders {
```

```
    https://dns.google/dns-query
};
```

Cohabitación sin loop

Damos nombre a nuestro servidor:

```
sudo vi /etc/hosts
sudo vi /etc/hostname
```

Configuramos nuestra red. Ver que **vamos a utilizar nuestro servidor como DNS**

```
sudo vi /etc/netplan/50-cloud-init.yaml
network:
    ethernets:
        enp0s3:
            dhcp4: false
            addresses:
                - 192.168.1.39/24
            routes:
                - to: default
                via: 192.168.1.1
            nameservers:
                addresses: [192.168.1.39]
    version: 2
    renderer: networkd
```

Ver el "**-4**" para que sólo trabajemos con ipv4

```
sudo vi /etc/default/named
aor@dnsserver:~$ sudo cat /etc/default/named
#
# run resolvconf?
RESOLVCONF=no

# startup options for the server
# #AOR#
OPTIONS="-u bind -4"
```

A partir de ahora lo más importante es que no tengamos peticiones circulares, entre bind9 y system-resolved y DNSStubListener = no

```
sudo vi /etc/systemd/resolved.conf
[Resolve]
DNS=8.8.8.8 1.1.1.1
FallbackDNS=8.8.4.4 1.0.0.1
#Domains=
DNSSEC=no
DNSOverTLS=no
MulticastDNS=no
LLMNR=no
#Cache=no-negative
#CacheFromLocalhost=no
DNSStubListener=no
#DNSStubListenerExtra=
#ReadEtcHosts=yes
#ResolveUnicastSingleLabel=no
#StaleRetentionSec=0
```

Ahora configuramos nuestro bind9

```
sudo vi /etc/bind/named.conf.options
aor@dnsserver:~$ cat /etc/bind/named.conf.options
options {
    directory "/var/cache/bind";
    recursion yes;  // Permitir la recursión
    allow-query { any; };  // Permitir consultas desde
cualquier IP
    forwarders {
        80.58.61.250;
        80.58.61.254;
    };
    //auth-nxdomain no;  // Conformarse al estándar
RFC1035
    listen-on { any; };  // Escuchar en todas las
interfaces de red
};
```

```
aor@dnsserver:~$ ls -la /etc/resolv.conf
lrwxrwxrwx 1 root root 39 ago 27 14:21 /etc/resolv.conf
-> ../run/systemd/resolve/stub-resolv.conf
aor@dnsserver:~$ sudo rm -f /etc/resolv.conf
[sudo] password for aor:
```

```
aor@dnsserver:~$ sudo touch /etc/resolv.conf
```

```
aor@dnsserver:~$ sudo vi /etc/resolv.conf
nameserver 192.168.1.39
search .
```

Y tenemos en

```
resolvectl
Global
           Protocols: -LLMNR -mDNS -DNSOverTLS
DNSSEC=no/unsupported
    resolv.conf mode: uplink
         DNS Servers: 8.8.8.8 1.1.1.1
Fallback DNS Servers: 8.8.4.4 1.0.0.1

Link 2 (enp0s3)
    Current Scopes: DNS
         Protocols: +DefaultRoute -LLMNR -mDNS -
DNSOverTLS DNSSEC=no/unsupported
       DNS Servers: 192.168.1.39
```

Comprobaciones

```
aor@dnsserver:~$ dig www.google.com
; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>>
www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
43970
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE:
403a30933d160cb401000000673e51abcfa9f7ca6b5684ed (good)
```

```
;; QUESTION SECTION:
;www.google.com.                IN    A

;; ANSWER SECTION:
www.google.com.         256    IN    A    216.58.215.132

;; Query time: 1 msec
;; SERVER: 192.168.1.39#53(192.168.1.39) (UDP)
;; WHEN: Wed Nov 20 21:16:27 UTC 2024
;; MSG SIZE  rcvd: 87
```

```
aor@dnsserver:~$ dig www.google.com @127.0.0.1

; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>>
www.google.com @127.0.0.1
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
28132
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE:
fd4b0251e51a57da01000000673e51b8b658b9dc65362821 (good)
;; QUESTION SECTION:
;www.google.com.                IN    A

;; ANSWER SECTION:
www.google.com.         243    IN    A    216.58.215.132

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1) (UDP)
;; WHEN: Wed Nov 20 21:16:40 UTC 2024
;; MSG SIZE  rcvd: 87
```

```
aor@dnsserver:~$ dig www.google.com @8.8.8.8code>

; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>>
www.google.com @8.8.8.8
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
30618
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.google.com.            IN    A

;; ANSWER SECTION:
www.google.com.        157    IN    A    142.250.200.100

;; Query time: 27 msec
;; SERVER: 8.8.8.8#53(8.8.8.8) (UDP)
;; WHEN: Wed Nov 20 21:17:04 UTC 2024
;; MSG SIZE  rcvd: 59
```

Análisis de Tráfico DNS con tcpdump

El uso de tcpdump es fundamental para cualquier administrador de sistemas y experto en seguridad, ya que permite la inspección directa de paquetes en tránsito. Mientras que herramientas como `dig` muestran lo que el servidor responde, tcpdump muestra lo que realmente está ocurriendo en la interfaz de red.

1. Monitorización Básica de Consultas

Para inspeccionar de forma general las consultas DNS que entran o salen de la máquina, se utiliza el siguiente comando:

```
sudo tcpdump -n port 53
```

- **Nota técnica:** El parámetro `-n` es crítico; evita que tcpdump intente resolver las direcciones IP de los paquetes capturados, lo que generaría un tráfico DNS adicional "infinito" y dificultaría la lectura del análisis.

2. Captura por Interfaz Específica

Si el servidor tiene múltiples interfaces (como `enp0s3`), es necesario especificar cuál queremos auditar para confirmar que las solicitudes están siendo enviadas correctamente desde esa máquina específica:

```
sudo tcpdump -i enp0s3 -n
```

3. Filtrado Avanzado para Auditoría de Seguridad

En seguridad informática, a menudo necesitamos aislar el tráfico hacia un servidor específico para detectar posibles anomalías o verificar respuestas entrantes tras un cambio de configuración.

Podemos filtrar simultáneamente por puerto y host:

```
sudo tcpdump –i enp0s3 –n 'port 53 and host 8.8.4.4'
```

4. Diagnóstico de Capa de Transporte (UDP vs. TCP)

La mayoría del tráfico DNS ocurre sobre **UDP**. Sin embargo, las transferencias de zona (AXFR) y las respuestas de gran tamaño requieren **TCP**. Si tras realizar las capturas anteriores no se observan respuestas, es posible que el tráfico UDP esté bloqueado por un firewall.

Para verificar si el puerto 53 TCP está operativo y permitir que tcpdump capture esa sesión, se puede forzar el uso de TCP en la herramienta de consulta:

```
dig +tcp @8.8.4.4 www.google.com
```

5. Interpretación de Resultados para el Administrador

- **Captura vacía:** Si al ejecutar tcpdump no se visualiza ningún paquete durante una consulta activa, sugiere que la configuración de red del sistema (o de BIND9) impide que las solicitudes lleguen a la interfaz de salida.
- **Sección de autoridad:** Durante el análisis de tráfico, es vital observar si las respuestas capturadas contienen secciones adicionales o de autoridad, lo cual ayuda a identificar si estamos ante un servidor autoritativo o una respuesta de caché.

DNSmasq

DNSmasq es un software ligero que actúa como servidor de caché DNS, DHCP, y TFTP. Es un software que proporciona:

- Almacenamiento en caché para el sistema de nombres de dominio (DNS)
- Servidor de protocolo de configuración dinámica de host (DHCP)
- Enrutador de publicidad
- Funciones de arranque de red para pequeñas redes informáticas

Instalamos el servidor:

```
sudo apt install dnsmasq
```

Paramos el servicio si se está ejecutando:

```
 sudo systemctl stop dnsmasq.service
```

Tenemos que hacer que cohabiten dnsmasq y systemd-resolved y para ello.

```
aor@dnsserver:~$ more /etc/systemd/resolved.conf
[Resolve]
DNS=127.0.0.1
DNSStubListener=no
```

Importante es DNSStubListener = no para liberar el puerto 53 del stub resolver de systemd-resolved y que lo pueda utilizar dnsmasq.

Reiniciamos el servicio systemd-resolved para que aplique los cambios:

```

```

```
sudo systemctl restart systemd-resolved
```

Configuramos dnsmasq en el archivo /etc/dnsmasq.conf, realizaremos una copia de seguridad antes:

```
sudo cp /etc/dnsmasq.conf /etc/dnsmasq.conf.bak
```

Editamos el archivo /etc/dnsmasq.conf y añadimos la configuración básica:

```
sudo vi /etc/dnsmasq.conf

domain-needed
bogus-priv
no-resolv
server=1.1.1.1
server=8.8.4.4
interface=enp0s3
listen-address=127.0.0.1
cache-size=1000
```

Opciones que podemos habilitar:

- **domain-needed:** No reenvía nombres sin un punto (.) o un nombre de dominio calificado, útil para evitar que consultas locales erróneas salgan a los servidores públicos.
- **bogus-priv:** Evita que las consultas de resolución inversa para rangos de IP privados (como 192.168.x.x) se reenvíen a los servidores DNS públicos.
- **no-resolv:** Indica a dnsmasq que no lea el archivo /etc/resolv.conf, sino que use los servidores DNS especificados explícitamente en su propio archivo de configuración.
- **server:** Especifica los servidores DNS upstream a los que dnsmasq reenviará las consultas que no pueda resolver localmente (en este caso, Cloudflare y Google).
- **interface:** Define en qué interfaz de red escuchará

dnsmasq (reemplaza enp0s3 por el nombre de tu interfaz).

- **listen-address:** Especifica las direcciones IP en las que dnsmasq escuchará. Es seguro incluir 127.0.0.1 para consultas locales.
- **cache-size:** Define el número de nombres de dominio que dnsmasq mantendrá en caché para acelerar futuras consultas.

Iniciamos el servicio dnsmasq:

```
sudo systemctl start dnsmasq.service
```

Comprobamos el estado del servicio:

```
sudo systemctl status dnsmasq.service
```

Realizamos pruebas de resolución:

```
dig www.google.com
```

Verificamos que en la respuesta aparezca SERVER: 127.0.0.1#53 o la IP de tu servidor, lo que indica que dnsmasq está procesando la consulta.

DHCP Server con DNSmasq

Añadimos la configuración de DHCP en el archivo /etc/dnsmasq.conf:

```
dhcp-range=172.16.6.200,172.16.6.250,255.255.0.0,12h
dhcp-option=option:router,172.16.0.1
dhcp-option=option:dns-server,172.16.6.132
```

- **dhcp-range:** Define el rango de direcciones IP que el

servidor DHCP asignará a los clientes, la máscara de subred y el tiempo de concesión (lease time) de las direcciones (en este caso, 12 horas).
- **dhcp-option = option:router:** Especifica la dirección IP de la puerta de enlace predeterminada (gateway) que se enviará a los clientes.
- **dhcp-option = option:dns-server:** Especifica la dirección IP del servidor DNS que se asignará a los clientes (en este caso, la IP del propio servidor dnsmasq).

Reiniciamos el servicio dnsmasq para aplicar los cambios de DHCP:

```
sudo systemctl restart dnsmasq.service
```

Reservas de IP (Static Leases)

Si queremos asignar una IP fija a un dispositivo específico basándonos en su dirección MAC, añadimos en el archivo /etc/dnsmasq.conf:

```
dhcp-host=08:00:27:bc:3d:a2,ubuntu-
client,172.16.6.210,infinite
```

- **08:00:27:bc:3d:a2:** Es la dirección MAC de la tarjeta de red del cliente.
- **ubuntu-client:** Es el nombre de host (hostname) que se le asignará al dispositivo.
- **172.16.6.210:** Es la dirección IP fija reservada para este dispositivo.
- **infinite:** Define que el tiempo de concesión es ilimitado, por lo que la IP no cambiará ni expirará.

DNSmasq como enrutador de publicidad

(Bloqueo de publicidad)

Dnsmasq puede utilizarse para bloquear publicidad y rastreadores a nivel de red, redirigiendo los dominios de publicidad conocidos a una dirección IP no válida (como 0.0.0.0 o 127.0.0.1), un concepto conocido como **DNS Blackhole** o "agujero negro" DNS.

¿Cómo funciona?

Carga de listas de bloqueo

Los administradores pueden obtener listas de dominios utilizados por servidores de publicidad (por ejemplo, desde proyectos como Pi-hole o fuentes públicas).

Configuración en DNSmasq

Estos dominios se configuran en dnsmasq para que, en lugar de consultar a los servidores de internet, respondan inmediatamente localmente con una IP nula.

Ejemplo de configuración para bloquear un dominio publicitario en /etc/dnsmasq.conf:

```
address=/doubleclick.net/0.0.0.0
address=/ads.example.com/127.0.0.1
```

Cuando un dispositivo de la red intente cargar un anuncio de `doubleclick.net`, dnsmasq responderá que su IP es 0.0.0.0. El navegador del usuario no podrá conectar y simplemente no mostrará el anuncio, ahorrando ancho de banda y mejorando la privacidad.

Redirección de publicidad

Además de bloquear, un "enrutador de publicidad" podría redirigir las consultas DNS de dominios publicitarios a otro servidor. Por ejemplo:

Redirigirlos a un servidor web local que muestra una página en blanco o un mensaje específico.
Redirigirlos hacia un servicio que monitoree las solicitudes para análisis o auditoría.

Configuración red del servidor ejemplo

En el ejemplo se ha tenido la siguiente configuración:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      addresses:
        - 172.16.6.132/16
      routes:
        - to: default
          via: 172.16.0.1
      nameservers:
        addresses: [1.1.1.1, 8.8.4.4]
```

dnsmasq y DHCP

Dnsmasq puede actuar como servidor DHCP, integrando resolución DNS con asignación dinámica de direcciones IP. Archivo de configuración (`/etc/dnsmasq.conf` o un archivo en `/etc/dnsmasq.d/`).

```
# Habilitar el servidor DHCP
dhcp-range=192.168.1.50,192.168.1.100,12h

# Especificar la puerta de enlace predeterminada
(opcional)
dhcp-option=option:router,192.168.1.1

# Establecer un servidor DNS (puedes usar el propio
servidor dnsmasq o uno externo)
dhcp-option=option:dns-server,192.168.1.1

# Establecer el dominio para los clientes DHCP
(opcional)
domain=xaor.com

# Integrar con DNS y hosts (opcional pero recomendado)
expand-hosts
```

- dhcp-range = 192.168.1.50,192.168.1.100,12h: Define el rango de IPs para asignar (de 192.168.1.50 a 192.168.1.100) y su tiempo de concesión (12 horas).
- dhcp-option = option:router,192.168.1.1: Especifica la dirección IP de la puerta de enlace (generalmente el router).
- dhcp-option = option:dns-server,192.168.1.1: Indica el servidor DNS que se proporcionará a los clientes DHCP.

Agregar asignaciones estáticas (opcional):

```
dhcp-
```

```
host=00:11:22:33:44:55,192.168.1.10,ns1.xaor.com,infinite
```

- 00:11:22:33:44:55: Dirección MAC del cliente.
- 192.168.1.10: Dirección IP fija que se asignará.
- ns1.xaor.com: Nombre de host asociado.
- infinite: La concesión no expira.

Permitir que dnsmasq escuche en la red correcta:

```
interface=eth0
```

```
sudo systemctl restart dnsmasq
```

Verificar el funcionamiento del servidor DHCP:

```
sudo tail -f /var/log/syslog | grep dnsmasq
```

Integración DNS-DHCP Automática

A diferencia de configuraciones complejas que requieren llaves TSIG (como BIND9 + ISC-DHCP-Server), dnsmasq realiza una **integración nativa y automática**.

Cuando un cliente solicita una dirección IP, envía su nombre de host (hostname). Dnsmasq asocia inmediatamente ese nombre con la IP asignada en su base de datos interna y genera dinámicamente tanto el registro directo (A) como el inverso (PTR). No requiere configuración adicional de zonas ni actualizaciones dinámicas manuales.

Asignar nombres automáticamente a clientes DHCP: Si los clientes envían un nombre de host durante la solicitud DHCP, dnsmasq lo utiliza para resolver tanto:

- De nombre a IP (DNS directo, registros A/AAAA).

- De IP a nombre (DNS inverso, registros PTR).

Para habilitarlo, asegúrate de que `expand-hosts` y `domain=` estén configurados en `dnsmasq.conf`. Usa una configuración básica de DHCP como esta:

```
domain=xaor.com
expand-hosts
dhcp-range=192.168.1.50,192.168.1.100,12h
```

Tenemos que quitar también la entrada que teníamos cuando sólo queríamos DNS: `no-dhcp-interface=`. Los clientes DHCP que proporcionen su nombre de host en las solicitudes aparecerán automáticamente en el DNS.

Usar asignaciones estáticas para nombres específicos: Si necesitas control sobre los nombres asignados, puedes usar entradas `dhcp-host` en la configuración:

```
dhcp-
host=00:11:22:33:44:55,client1.xaor.com,192.168.1.10
```

Esto asigna la IP estática 192.168.1.10 al cliente con MAC 00:11:22:33:44:55.

Ventajas de la Integración DNS-DHCP en Dnsmasq

- **Simplicidad Extrema:** Ideal para enrutadores domésticos, redes pequeñas/medianas o entornos de laboratorio. No hay necesidad de mantener archivos de zonas separados que se sincronicen mediante scripts o mecanismos complejos.
- **Consistencia de Red:** Cualquier máquina conectada por DHCP puede ser localizada por su nombre inmediatamente por el resto de las máquinas de la red local sin necesidad de configurar DNS estáticos para cada una.

- **Bajo consumo de recursos:** Al ejecutarse en un solo proceso ligero, gestiona ambos servicios compartiendo la misma tabla de estados en memoria, lo que minimiza el uso de CPU y RAM.

master-primary

Primary DNS Server

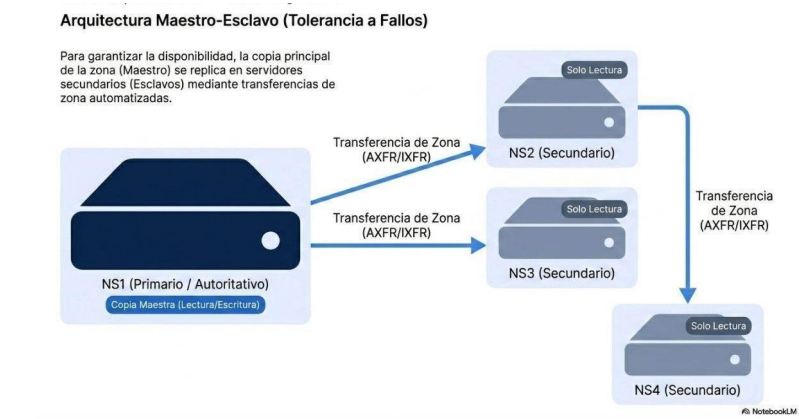
- El término "primary" generalmente se refiere al servidor DNS que tiene la autoridad para una zona de dominio específica.
- En una relación de servidor de nombres maestro y esclavo (master-slave), el servidor primario es aquel que mantiene la copia maestra (o principal) de la zona de dominio. Este servidor tiene la autoridad final y es responsable de realizar actualizaciones directas en la zona.
- El servidor primario es la fuente autorizada de la información de zona y es donde se realizan la mayoría de las actualizaciones y modificaciones.

Master DNS Server:

- El término "master" también se utiliza para referirse al servidor DNS principal en una relación de servidor maestro y esclavo.
- En este contexto, el servidor maestro es aquel que mantiene la copia principal y autorizada de la zona de dominio. Los servidores esclavos (también llamados secundarios) obtienen copias de la zona del servidor maestro a través de transferencias de zona.
- El servidor maestro es donde se realizan las actualizaciones directas y se considera la fuente primaria de la información de la zona.

Servidor DNS Master-Slave con Bind9

write_enable



En este post explicaremos como configurar un servidor DNS con la herramienta bind9 y crear un esclavo que se actualice automáticamente. Para esta práctica se utilizaran 2 sistemas Debian con una subred 192.168.0.0/24 siendo NS1 el servidor DNS maestro y el NS2 el servidor DNS esclavo.

Además de la instalación y configuración, se intentara securizar lo máximo posible para evitar posibles agujeros de seguridad.

NS1

Este servidor será el utilizado como maestro.

Actualizar repositorios

```
apt update
```

Instalar bind9

```
apt install bind9 bind9utils bind9-doc
```

Configurar bind9

```
vim /etc/bind/named.conf.options
```

```
```

```
acl "trusted" {
        192.168.0.104;          # ns1 (IP servidor bind)
        192.168.0.105;          # ns2
        192.168.0.0/24;         # subred que
utilizaremos
};

options {
        directory "/var/cache/bind";

        recursion yes;                  # Permitir la
recursión
        allow-recursion { trusted; };  # Permitir la
recursión solo a los hosts declarados
        allow-query { trusted; }        # Permitir las
consultas solo a los hosts declarados
        dnssec-validation auto;         # Auto-validación
de DNSSEC
        listen-on { 192.168.0.104; };  # IP de escucha
        allow-transfer { none; };       # No permitir la
transferencia
        version "DNS server";           # Ocultar la
versión de bind
        forwarders {
                8.8.8.8;         #Google DNS
                209.67.222.222; #OpenDNS
        };

        auth-nxdomain no;    # conform to RFC1035
};
```

Nota: **acl** significa **Access Control List**. Una ACL (Lista de Control de Acceso) es una herramienta que se utiliza para definir reglas que especifican qué entidades están permitidas o denegadas para realizar ciertas acciones.
En el caso de BIND and DNS, las ACL se utilizan comúnmente para controlar el acceso a las zonas y definir políticas de seguridad.

recursion yes;

Permite que el servidor DNS realice búsquedas recursivas en nombre de los clientes. Cuando un servidor realiza una búsqueda recursiva, intenta obtener la respuesta completa consultando otros servidores DNS si es necesario.

allow-recursion { trusted; };

Restringe la capacidad de realizar búsquedas recursivas a los hosts que se han declarado como "trusted" (de confianza). Esto ayuda a evitar que el servidor DNS sea utilizado para realizar consultas recursivas por clientes no autorizados.
La declaración que asigna direcciones IP al conjunto "trusted" es algo como:

```
acl trusted {
    192.168.0.1;
    192.168.0.2;
    192.168.0.3;
    # ... otras direcciones IP confiables ...
    192.168.0.0/24;  # Subred confiable. Tiene
sentido tener subredes en las que confías
};
```

Después de definir el conjunto "trusted", la configuración de allow-recursion se refiere a este conjunto de direcciones IP de confianza.

allow-query { trusted; }
Restringe la capacidad de realizar consultas DNS al mismo conjunto de hosts "trusted" que se han especificado. Esto limita quiénes pueden realizar consultas al servidor DNS.

dnssec-validation auto;
Habilita la auto-validación de DNSSEC (Domain Name System Security Extensions).
DNSSEC proporciona seguridad adicional al sistema de nombres de dominio al verificar la autenticidad y la integridad de las respuestas DNS.
Ver más adelante

listen-on { 192.168.0.104; };
Especifica la dirección IP en la que el servidor DNS escuchará las consultas DNS. En este caso, el servidor escuchará en la dirección IP 192.168.0.104.
Si tenemos más de un interface de red y queremos que escuche en todos pondríamos 0.0.0.0

allow-transfer { none; };
No permite transferencias de zona desde este servidor. Las transferencias de zona son operaciones que permiten que

otro servidor DNS copie la información de zona desde este servidor. Al configurarlo como "none", se evita que otros servidores realizen transferencias de zona desde este.

version "DNS server";
Define una cadena que se incluirá en las respuestas del servidor DNS. En este caso, la cadena es "DNS server". Esto puede utilizarse para ocultar la versión específica del software BIND que se está ejecutando.

forwarders { 8.8.8.8; 209.67.222.222; };
Especifica servidores DNS a los cuales se enviarán las consultas que el servidor DNS no puede responder localmente. En este caso, se han configurado servidores DNS de Google (8.8.8.8) y OpenDNS (209.67.222.222) como servidores forwarders.

auth-nxdomain no;
Indica al servidor DNS que no autentique las respuestas negativas (NXDOMAIN) mediante DNSSEC. Esta configuración se ajusta a la especificación RFC1035.

Definir las Zonas, en este caso se le indica que son maestras

```
vim /etc/bind/named.conf.local
```

```
zone "micasa.com" {
 type master;
 file "/etc/bind/zones/db.micasa.com"; # Path del
fichero de zona directa
 allow-transfer { 192.168.0.105; };       # Permitir la
transferencia a NS2
};

zone "0.168.192.in-addr.arpa" {
 type master;
 file "/etc/bind/zones/db.192.168.0"; # Path del fichero
de zona inversa - subred:192.168.0.0/24
 allow-transfer { 192.168.0.105; };    # Permitir la
transferencia a NS2
};
```

Comprobar la sintaxis de los named.conf

```

```

```
named-checkconf
```

Zonas

Crear el directorio de las zonas

```
mkdir /etc/bind/zones
```

Zona directa

Crear la zona a partir del archivo de ejemplo

```
cp /etc/bind/db.local /etc/bind/zones/db.micasa.com
```

Editar el archivo de zona directo. En este archivo hay que realizar varios cambios como el SOA, los NS, los registros y el **serial** cada vez que se edita

```
vim /etc/bind/zones/db.micasa.com
```

```
$TTL    604800
@       IN      SOA     micasa.com. ismael.micasa.com. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache
TTL

; NS records
                        IN      NS      ns1.micasa.com.
                        IN      NS      ns2.micasa.com.

; name servers - A records
ns1.micasa.com.       IN      A       192.168.0.104
ns2.micasa.com.       IN      A       192.168.0.105

; host - A records
host1.micasa.com.     IN      A       192.168.0.106
```

```
host2.micasa.com.      IN      A       192.168.0.107
```

Comprobar la sintaxis de la zona directa

```
named-checkzone micasa.com /etc/bind/zones/db.micasa.com

/etc/bind/zones/db.micasa.com:17: ignoring out-of-zone
data (ns1.micasa.com)
/etc/bind/zones/db.micasa.com:18: ignoring out-of-zone
data (ns2.micasa.com)
zone micasa.com/IN: loaded serial 3
OK
```

Zona Inversa

Crear la zona a partir del archivo de ejemplo

```
cp /etc/bind/db.127 /etc/bind/zones/db.192.168.0
```

Editar el archivo de zona inversa. En este archivo hay que realizar varios cambios como el SOA, los NS, los registros y el **serial** cada vez que se edita

```
vim /etc/bind/zones/db.192.168.0
```

```
$TTL    604800
@       IN      SOA     micasa.com. ismael.micasa.com. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache
TTL
;
; NS records
        IN      NS      ns1.micasa.com.
        IN      NS      ns2.micasa.com.

; PTR Records
104     IN      PTR     ns1.micasa.com.
105     IN      PTR     ns2.micasa.com.
```

```
106     IN      PTR     host1.micasa.com.
107     IN      PTR     host2.micasa.com.
```

Comprobar la sintaxis de la zona inversa

```
named-checkzone 0.168.192.in-addr-arpa /etc/bind/zones/
db.192.168.0
zone 0.168.192.in-addr-arpa/IN: loaded serial 3
OK
```

Cuando todas las comprobaciones hayan sido exitosas, reiniciar bind

```
systemctl restart bind9
```

NS2

Actualizar repositorios

```
apt update
```

Instalar bind9

```
apt install bind9 bind9utils bind9-doc
```

Editar la configuración de bind9

```
vim /etc/bind/named.conf.options
```

```
acl "trusted" {
        192.168.0.105;          # ns2 (IP servidor
secundario bind)
        192.168.0.104;          # ns1
        192.168.0.0/24;
};

options {
        directory "/var/cache/bind";
```

```
        recursion yes;
        allow-recursion { trusted; };      # Permitir la
recursión a los hosts definidos
        allow-query { trusted; }           # Permitir
consultas a los hosts definidos
        dnssec-validation auto;            # Auto-
validación de DNSSEC
        listen-on { 192.168.0.105; };      # ns2 private
IP address
        allow-transfer { none; };          #
deshabilitar la transferencia
        version "DNS server";              # Ocultar la
versión del servidor DNS
        forwarders {
                8.8.8.8;         #Google DNS
                209.67.222.222; #OpenDNS
        };

        auth-nxdomain no;    # conform to RFC1035
};
```

Definir las zonas e indicar que son de tipo slave. Una vez con esto. Cuando se reinicie el servicio automáticamente se sincronizará con el master

```
vim /etc/bind/named.conf.local
```

```
zone "micasa.com" {
 type slave;
 file "/var/cache/bind/db.micasa.com";
 masters { 192.168.0.104; }; # IP del NS1
};

zone "0.168.192.in-addr.arpa" {
 type slave;
 file "/var/cache/bind/db.192.168.0";
 masters { 192.168.0.104; }; # IP del NS1
};
```

Comprobar la sintaxis de los named.conf

```
named-checkconf
```

Reiniciar bind9

```
systemctl restart bind9
```

Seguridad

Directivas

- auth-nxdomain no;

DNSSEC

DNSSEC añade una capa de seguridad adicional al protocolo DNS que permite comprobar la integridad y autenticidad de los datos. Gracias a estas extensiones de seguridad se pueden prevenir ataques de suplantación y falsificación.

Las nuevas funcionalidades del protocolo están basadas en criptografía asimétrica, también conocida como criptografía de clave pública. Cuando un usuario realiza una búsqueda de un dominio que cuenta con DNSSEC, en el proceso se envía la información necesaria para resolver la búsqueda, es decir, la dirección IP del dominio que se está buscando. Pero también se envían varias las firmas de las claves asociadas a los distintos servidores DNS que se han consultado.

Recursividad sin forwarders

Cuando un servidor DNS no tiene configurados forwarders, realiza la resolución recursiva buscando la información directamente desde las raíces del sistema de nombres de dominio (DNS). Proceso general de cómo se realiza la recursión en un servidor DNS sin forwarders:

1. **Solicitud Inicial:** Un cliente hace una solicitud al servidor DNS para resolver un nombre de dominio.
2. **Búsqueda en Caché Local:** El servidor DNS verifica su caché local para ver si ya ha resuelto esa consulta recientemente. Si la información está en la caché y aún es válida, se devuelve al cliente.
3. **Consulta a la Zona Autoritativa:** Si la información no está en la caché o está caducada, el servidor DNS realiza una consulta a la zona autoritativa del dominio correspondiente. Comienza consultando a los servidores raíz para obtener información sobre los servidores de nombres autoritativos para el dominio de nivel superior (TLD) del nombre de dominio.
4. **Consulta a Servidores de Nombres de Nivel Superior (TLD):** Después de obtener información de los servidores raíz, el servidor DNS hace consultas a los servidores de nombres de nivel superior (TLD) para obtener información sobre los servidores de nombres autoritativos para el dominio de segundo nivel.
5. **Consulta a Servidores Autoritativos del Dominio:** Continuando con el proceso, el servidor DNS consulta a los servidores autoritativos del dominio específico para obtener la información de resolución final.
6. **Caché y Respuesta al Cliente:** Una vez que la información es obtenida, se almacena en la caché local del servidor DNS para futuras consultas y se envía la respuesta al cliente que realizó la solicitud inicial.

Este proceso de resolución recursiva puede ser intensivo en términos de tiempo y recursos, ya que implica múltiples consultas a diferentes niveles del sistema de nombres de dominio. Configurar forwarders puede ayudar a acelerar el proceso al dirigir las consultas directamente a servidores DNS específicos en lugar de seguir el proceso completo de resolución recursiva.

RNDC

En BIND9 (Berkeley Internet Name Domain), rndc es una herramienta de administración que permite controlar el servidor DNS de forma remota o local.

Las siglas rndc significan Remote Name Daemon Control.

Importante: Usa un sistema de autenticación basado en claves compartidas (shared secret) para garantizar comunicaciones seguras entre el cliente rndc y el servidor BIND.

Usos:

- **Recarga de configuración o zonas** (no hace falta reiniciar el servicio):

  ```
  rndc reload
  ```

  Para recargar una zona específica:

  ```
  rndc reload nombre_de_zona
  ```

  Reinicio controlado del servidor DNS:

  ```
  rndc restart
  ```

  Detener o iniciar el servidor (Detiene el servidor BIND):

  ```
  rndc stop
  ```

- **Gestión de caché:**

Limpia la caché del servidor para eliminar registros desactualizados:

```
rndc flush
```

Para una zona específica:

```
rndc flushname nombre_de_zona
```

- **Ver el estado del servidor:**

  Proporciona información sobre el estado actual del servidor:

  ```
  rndc status
  ```

- **Habilitar o deshabilitar funciones:**

  Deshabilitar temporalmente respuestas automáticas de zonas:

  ```
  rndc freeze
  ```

  Volver a habilitarlas:

  ```
  rndc thaw
  ```

- **Administración de estadísticas:**

  Generar y visualizar estadísticas:

  ```
  rndc stats
  ```

Requisitos para usar rndc

El archivo de configuración `rndc.conf` o las opciones incluidas en `named.conf` configuran la autenticación.

Ejemplo de una clave en `named.conf`:

```
key "rndc-key" {
    algorithm hmac-sha256;
    secret "clave_secreta_compartida_en_base64==";
};

controls {
    inet 127.0.0.1 port 953
    allow { 127.0.0.1; } keys { "rndc-key"; };
};
```

Ficheros ejemplo

> ### *Nota de Seguridad: Transferencias de Zona y TSIG*
>
> *Aunque en el siguiente ejemplo de named.conf.local se restringen las transferencias a una dirección IP específica (192.168.1.73), este método es vulnerable al IP Spoofing (suplantación de IP).*
> *En entornos de producción, se recomienda sustituir las ACLs basadas en IP por TSIG (Transaction Signature).*
> *Al utilizar TSIG, el servidor primario solo permitirá la transferencia (AXFR) si el servidor secundario presenta una clave secreta compartida válida, garantizando así la autenticidad e integridad de los datos transferidos.*
> *Para implementar esta mejora, la directiva en el bloque zone debería cambiarse a:*
>
> ```
> allow-transfer { key "nombre_de_tu_clave"; };
> ```
>
> *(Ver sección de "Seguridad Avanzada > TSIG" para el proceso de generación de claves con tsig-keygen)*

```
::::::::::::::
named.conf
::::::::::::::
// This is the primary configuration file for the BIND
DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for
information on the
// structure of BIND configuration files in Debian,
*BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/
bind/named.conf.local
```

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
//include "/etc/bind/named.conf.default-zones";

::::::::::::::
named.conf.local
::::::::::::::
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not
used in your
// organization
//include "/etc/bind/zones.rfc1918";
//AOR//----------------------------------------
zone "aor.com" {
    type primary;
    file "/etc/bind/zones/db.aor.com"; # zone file path
    allow-transfer { 192.168.1.73; };           # ns2
private IP address - secondary
};

// Como estoy en 192.168.1.0/24 tenemos
zone "1.168.192.in-addr.arpa" {
    type primary;
    file "/etc/bind/zones/db.1.168.192";  #
192.168.1.0/24 subnet
    allow-transfer { 192.168.1.73; };  # ns2 private IP
address - secondary
};
//
zone "0.168.192.in-addr.arpa" {
    type primary;
    file "/etc/bind/zones/db.0.168.192"; #
192.168.0.0/24 subnet
    allow-transfer { 192.168.1.73; };
};

//AOR/----------------------------------------
::::::::::::::
named.conf.options
::::::::::::::
acl trusted {
        192.168.0.114;
        192.168.1.73;
```

```
        192.168.0.71;
        192.168.1.74;
};
options {
 directory "/var/cache/bind";

 // If there is a firewall between you and nameservers
you want
 // to talk to, you may need to fix the firewall to
allow multiple
 // ports to talk.  See http://www.kb.cert.org/vuls/
id/800113

 // If your ISP provided one or more IP addresses for
stable
 // nameservers, you probably want to use them as
forwarders.
 // Uncomment the following block, and insert the
addresses replacing
 // the all-0's placeholder.

 //AOR//----------------------------
 recursion yes;                 # enables recursive
queries
        allow-recursion { trusted; };  # allows
recursive queries from "trusted" clients
        listen-on { 192.168.0.114; };   # ns1 private IP
address - listen on private network only
        allow-transfer { none; };      # disable zone
transfers by default
 allow-query { trusted; };
 minimal-responses yes;
 forwarders { 8.8.8.8; 8.8.4.4; };
 empty-zones-enable no;  # https://serverfault.com/
questions/833824/how-do-you-configure-bind-to-forward-
reverse-dns-queries-to-another-dns-server
 //AOR//----------------------------

 //
================================================================
 // If BIND logs error messages about the root key being
expired,
 // you will need to update your keys.  See https://
www.isc.org/bind-keys
 //
================================================================
 dnssec-validation auto;
```

```
 //AOR//--------------------
 // listen-on-v6 { any; };
 //listen-on-v6 { none; };
 //allow-transfer { none; };
 auth-nxdomain no;    # conform to RFC1035
 //AOR//-------------------
};

::::::::::::::
db.0.168.192
::::::::::::::
;
; BIND reverse data file for local loopback interface
;
$TTL 604800
@ IN SOA aor.com. admin.aor.com. (
         7  ; Serial
    604800  ; Refresh
     86400  ; Retry
   2419200  ; Expire
    604800 ) ; Negative Cache TTL
;
; name servers - NS records
      IN      NS      ns1.aor.com.
      IN      NS      ns2.aor.com.
; PTR Records
114 IN      PTR     ns1.aor.com.
71 IN      PTR     kali00.aor.com.

::::::::::::::
db.1.168.192
::::::::::::::
;
; BIND reverse data file for local loopback interface
;
$TTL 604800
@ IN SOA aor.com. admin.aor.com. (
         7  ; Serial
    604800  ; Refresh
     86400  ; Retry
   2419200  ; Expire
    604800 ) ; Negative Cache TTL
;
; name servers - NS records
      IN      NS      ns1.aor.com.
      IN      NS      ns2.aor.com.
; PTR Records
```

```
73   IN      PTR     ns2.aor.com.
74 IN      PTR     host1.aor.com.
71 IN      PTR     kali.aor.com.

::::::::::::::
db.aor.com
::::::::::::::
;
; BIND data file for local loopback interface
;
;AOR--- reemplazamos el localhost por el ns1 y el root
por admin
; Cada vez que cambiamos incrementamos el serial en +1
$TTL 604800
@ IN SOA aor.com. admin.aor.com. (
         9  ; Serial
    604800  ; Refresh
     86400  ; Retry
   2419200  ; Expire
    604800 ) ; Negative Cache TTL
; name servers - NS records
    IN      NS      ns1.aor.com.
    IN      NS      ns2.aor.com.
; name servers - A records
ns1          IN      A       192.168.0.114
ns2          IN      A       192.168.1.73
; 192.168.0.0/24 - A records
host1        IN      A      192.168.1.74
kali        IN      A      192.168.1.71
kali00  IN A 192.168.0.71
;
www IN CNAME ns1.aor.com.
editorial IN CNAME ns1.aor.com.
```

DNS Verifica

```
sudo rndc status
version: BIND 9.18.28-0ubuntu0.24.04.1-Ubuntu (Extended
Support Version) <id:>
running on localhost: Linux x86_64 6.8.0-45-generic #45-
Ubuntu SMP PREEMPT_DYNAMIC Fri Aug 30 12:02:04 UTC 2024
boot time: Mon, 25 Nov 2024 17:37:48 GMT
last configured: Mon, 25 Nov 2024 17:37:48 GMT
configuration file: /etc/bind/named.conf
CPUs found: 1
worker threads: 1
UDP listeners per interface: 1
number of zones: 103 (98 automatic)
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/900/1000
tcp clients: 0/150
TCP high-water: 0
server is up and running
```

Instala y utiliza tcpdump para inspeccionar las consultas DNS salientes:

```
sudo tcpdump -n port 53
```

Si tcpdump no ha capturado ningún paquete DNS sugiere que las solicitudes no están siendo enviadas correctamente desde la máquina. Realiza una captura más amplia para confirmar:

```
sudo tcpdump -i enp0s3 -n
```

Realiza otra captura con tcpdump mientras ejecutas una consulta con dig (por ejemplo al dns 8.8.4.4). Busca específicamente respuestas entrantes:

```

```

```
sudo tcpdump -i enp0s3 -n 'port 53 and host 8.8.4.4'
```

Si no tenemos respuestas vía udp puede que DNS sobre TCP no esté bloqueado. Prueba ejecutar:

```
dig +tcp @8.8.4.4 www.google.com
```

Verifica puerto

tcp

```
nc -zv <IP_del_servidor> 53
```

- -z: Escanea el puerto sin enviar datos.
- -v: Modo detallado para mostrar mensajes sobre la conexión.

Salida

```
Connection to 192.168.1.100 53 port [tcp/domain]
succeeded!
```

Salida esperada si está cerrado o inaccesible:

```
nc: connect to 192.168.1.100 port 53 (tcp) failed:
Connection refused
```

udp

```
nc -zvu <IP_del_servidor> 53
```

- -u: Especifica que el protocolo es UDP.
- -z: Escanea el puerto.

- -v: Modo detallado.

Salida

```
Connection to 192.168.1.100 53 port [udp/domain]
succeeded!
```

Salida esperada si está inaccesible:

```
nc: connect to 192.168.1.100 port 53 (udp) failed:
Connection refused
```

DNS. Forwarders y recursión

Nota

Para las pruebas en caso que tengas cohabitando systemd-resolved y bind9 tenemos que parar el resolutor primero:

```
sudo systemctl stop system-resolved
sudo systemctl disable system-resolved
```

y activarlo una vez finalizado el ejercicio:

```
sudo systemctl enable system-resolved
sudo systemctl start system-resolved
```

Sin forwarders

Caso práctico:

- Tenemos nuestro propio servidor DNS y no le añadimos forwarders.
- Tenemos un cliente que tiene configurado como DNS primario tu servidor y como secundario un DNS externo como 1.1.1.1.

Pasos en la resolución:

Resolución de nombres

- Resolución desde tu servidor DNS:

Nuestro servidor responderá únicamente para las zonas para las que es autoritativo (es decir, las zonas locales configuradas en micasa.com). Dado que no tiene forwarders ni la recursión configurada para buscar en los servidores raíz públicos, cualquier consulta para un dominio externo (como google.com) fallará o será rechazada.

- Fallo en la resolución externa:
  Dado que tu servidor DNS no puede resolver google.com, el cliente experimentará un retraso o recibirá una respuesta de error (como SERVFAIL o NXDOMAIN) desde tu servidor.
- Paso al DNS secundario:
  Al no recibir una respuesta válida del DNS primario, el sistema operativo del cliente (o el navegador) cambiará de servidor e intentará realizar la misma consulta utilizando el DNS secundario configurado (1.1.1.1).
- Resolución exitosa desde el secundario:
  El servidor DNS secundario (1.1.1.1), que es un resolutor público global, procesará la consulta de forma normal, buscará los registros de google.com en internet y devolverá la dirección IP correcta al cliente.

Ventajas

- Mayor control y privacidad, ya que no dependes de un intermediario (forwarder) que registre todas tus consultas.
- Respuestas directas desde la fuente autoritativa, minimizando el riesgo de envenenamiento de caché (cache poisoning) en servidores intermedios de terceros.
- Evitas el "baking" de consultas a un forwarder, lo que podría reducir el rastro de datos que dejan tus consultas en terceros.

Desventajas

- Mayor carga en el servidor, más recursos (CPU,

memoria y ancho de banda) en comparación con reenviar consultas a un forwarder.
- Menor optimización, no aprovechamos la optimización de los proveedores de DNS (como Google o Cloudflare).
- Bloqueos regionales o filtros, ciertos ISPs o gobiernos podrían restringir el acceso a los servidores raíz o a algunos servidores autoritativos, lo que puede dificultar la resolución directa.

Acerca de la recursión

Habilitar la recursión en BIND9

```
options {
    recursion yes;
    // otras opciones...
};
```

Limitar el acceso a la recursión a ciertos clientes

Permitir la recursión solo a ciertas direcciones IP (por ejemplo, las de tus servidores internos o clientes confiables):

```
acl "trusted-clients" {
    192.168.1.0/24;  // Rango de IP de clientes
confiables, por ejemplo, tu red interna
    // Agrega otras IPs o rangos si es necesario
};
```

```
options {
    recursion yes;
    allow-recursion { trusted-clients; };
    // otras opciones...
};
```

Deshabilitar la recursión para el resto de los clientes

```
options {
    recursion yes;
    allow-recursion { trusted-clients; };
    allow-query { any; }; // Permitir consultas no
recursivas (autoritativas) a cualquiera
};
```

DNS Dinámico (DDNS)

Tu Dirección Permanente en un Mundo Cambiante

Mientras que el DNS estándar está diseñado para la estabilidad de las empresas, el **DDNS** está diseñado para la flexibilidad y el acceso remoto en entornos con IPs variables.

1. ¿Qué es el DDNS?

DDNS significa **Sistema Dinámico de Nombres de Dominio**. Para entenderlo, primero debemos recordar que el DNS estándar es como una guía telefónica que une un nombre (ej. `youtube.com`) con un número (su dirección IP).

2. El problema: Direcciones IP Estáticas vs. Dinámicas

En el mundo profesional (como Google o YouTube), los servidores utilizan **direcciones IP estáticas**, lo que significa que su "número" nunca cambia. Esto es fundamental porque si la IP de YouTube cambiara, el DNS tradicional podría tardar hasta **24 horas** en actualizar su base de datos, dejando el sitio inaccesible durante todo ese tiempo.

Sin embargo, en nuestras casas, los proveedores de internet (ISP) nos asignan **direcciones IP dinámicas** mediante un servidor DHCP. Esto significa que tu dirección IP cambia periódicamente sin previo aviso.

3. El reto de la administración remota

Como administradores de sistemas, a veces necesitamos acceder a dispositivos en el hogar (una computadora de trabajo, una cámara de seguridad o un enrutador) desde cualquier parte del mundo.

- **Sin DDNS:** Si intentas conectarte usando la dirección IP que anotaste ayer y tu ISP la cambió hoy, no podrás entrar porque estarás llamando a un "número antiguo".
- **Con DDNS:** Aquí es donde entra la solución.

4. ¿Cómo funciona la solución DDNS?

El DNS dinámico te permite crear un **nombre de host personalizado** (por ejemplo: `mi-casa.ddns.org`) y vincularlo a tu dispositivo.

- **Automatización:** Si la dirección IP de tu casa cambia, el servicio DDNS detecta el cambio y actualiza automáticamente el nombre de host con la nueva IP.
- **Acceso simplificado:** No importa cuántas veces cambie tu IP; tú siempre escribirás `mi-casa.ddns.org` en tu navegador o cliente de escritorio remoto para conectarte.

DNS

```
$ cat /etc/bind/named.conf

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
//include "/etc/bind/named.conf.default-zones";
```

```
$ cat /etc/bind/named.conf.options
options {
 directory "/var/cache/bind";

 forwarders {
  8.8.8.8;
  8.8.4.4;
 };

 dnssec-validation auto;

 listen-on-v6 { any; };
```

```
};
```

```
$ cat /etc/bind/named.conf.local

zone "aorsrv.com" {
    type primary;
    file "/var/cache/bind/zones/db.aorsrv.com";
    allow-update { any; };
};

// Como estoy en 192.168.1.0/24 tenemos
zone "2.168.192.in-addr.arpa" {
    type primary;
    file "/var/cache/bind/zones/db.192.168.2.0";
    allow-update { any; };
};
```

```
aor@ubusrv11:~$ tree /var/cache/bind/
/var/cache/bind/
    managed-keys.bind
    zones
        db.192.168.2.0
        db.aorsrv.com
```

```
ls -la /var/cache/bind/zones/
total 24
drwxr-xr-x 2 bind bind 4096 dic 18 08:44 .
drwxrwxr-x 3 root bind 4096 dic 18 08:37 ..
-rw-r--r-- 1 bind bind  285 dic 18 08:20 db.192.168.2.0
-rw-r--r-- 1 bind bind  322 dic 18 08:30 db.aorsrv.com
```

```
$ cat /var/cache/bind/zones/db.aorsrv.com
;
; BIND data file for local loopback interface
;
$TTL 604800
@ IN SOA aorsrv.com. root.aorsrv.com. (
         2  ; Serial
    604800  ; Refresh
     86400  ; Retry
   2419200  ; Expire
```

```
    604800 ) ; Negative Cache TTL
;
@ IN NS ns1.aorsrv.com.
ns1.aorsrv.com.  IN A 192.168.2.11
;
ubusrv11.aorsrv.com. IN A 192.168.2.11
```

```
$ cat /var/cache/bind/zones/db.192.168.2.0
;
; BIND reverse data file for local loopback interface
;
$TTL 604800
@ IN SOA aorsrv.com. root.aorsrv.com. (
         1  ; Serial
    604800  ; Refresh
     86400  ; Retry
   2419200  ; Expire
    604800 ) ; Negative Cache TTL
;
@ IN NS ns1.aorsrv.com.
11 IN PTR ubusrv11.aorsrv.com.
```

DCHP

```
$ cat /etc/dhcp/dhcpd.conf
# dhcpd.conf

# option definitions common to all supported networks...
option domain-name "aorsrv.com";
option domain-name-servers ns1.aorsrv.com,
ns2.aorsrv.com;

default-lease-time 600;
max-lease-time 7200;

# Configuración de Dynamic DNS (DDNS)
ddns-updates on;
ddns-update-style interim;
update-static-leases on;

# Zona para el dominio principal
zone aorsrv.com. {
  primary 192.168.2.11;
```

```
}

# Zona inversa para la subred
zone 2.168.192.in-addr.arpa. {
  primary 192.168.2.11;
}

# Configuración para una subred interna
subnet 192.168.2.0 netmask 255.255.255.0 {
  range 192.168.2.200 192.168.2.250;
  option domain-name-servers 192.168.2.11;
  option domain-name "aorsrv.com";
  option subnet-mask 255.255.255.0;
  option routers 192.168.2.1;

  # Configuración de DDNS específica para la subred
  ddns-rev-domainname "in-addr.arpa.";

  # Otras opciones DHCP
  default-lease-time 600;
  max-lease-time 7200;
}
```

Para ddns-updates y ddns-update-style

```
ddns-updates on;
```

Opciones para ddns-update-style:

- ad-hoc: Se permite la actualización dinámica solo si el cliente lo solicita explícitamente. las actualizaciones dinámicas de DNS solo se permiten si el cliente DHCP lo solicita explícitamente. Esto significa que el cliente debe estar configurado para enviar una solicitud de actualización de DNS al servidor DHCP.
- interim: Similar a ad-hoc, pero también permite actualizaciones de clientes que no pueden realizar una actualización completa.
- none: Deshabilita completamente la actualización dinámica.
- standard: Permite la actualización dinámica de acuerdo con la especificación estándar de DHCP (Recomendado para la mayoría de los casos).

```

```

```
ddns-update-style standard;
```

Consideraciones

15

```
# Set to yes to apply rules to support IPv6 (no means only IPv6 on loopback
# accepted). You will need to 'disable' and then 'enable' the firewall for
# the changes to take affect.
IPV6=yes

# Set the default input policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_INPUT_POLICY="DROP"

# Set the default output policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_OUTPUT_POLICY="ACCEPT"

# Set the default forward policy to ACCEPT, DROP or REJECT.  Please note that
# if you change this you will most likely want to adjust your rules
###AOR#DEFAULT_FORWARD_POLICY="DROP"
DEFAULT_FORWARD_POLICY="ACCEPT"

# Set the default application policy to ACCEPT, DROP, REJECT or SKIP. Please
# note that setting this to ACCEPT may be a security risk. See 'man ufw' for
# details
DEFAULT_APPLICATION_POLICY="SKIP"

# By default, ufw only touches its own chains. Set this to 'yes' to have ufw
# manage the built-in chains too. Warning: setting this to 'yes' will break
# non-ufw managed firewall rules
MANAGE_BUILTINS=no

#
# IPT backend
#
# only enable if using iptables backend
IPT_SYSCTL=/etc/ufw/sysctl.conf

# Extra connection tracking modules to load. IPT_MODULES should typically be
# empty for new installations and modules added only as needed. See
# 'CONNECTION HELPERS' from 'man ufw-framework' for details. Complete list can
# be found in net/netfilter/Kconfig of your kernel source. Some common modules:
# nf_conntrack_irc, nf_nat_irc: DCC (Direct Client to Client) support
aor@ns1:~$ more /etc/default/ufw
```

15. Si tenemos firewall
Tenemos que establecer como política por defecto para el forward el ACCEPT, es el que va a permitir que podamos enrutar la información.

16

```
#
# only enable if using iptables backend
IPT_SYSCTL=/etc/ufw/sysctl.conf

# Extra connection tracking modules to load. IPT_MODULES should typically b
# empty for new installations and modules added only as needed. See
# 'CONNECTION HELPERS' from 'man ufw-framework' for details. Complete list
# be found in net/netfilter/Kconfig of your kernel source. Some common modu
# nf_conntrack_irc, nf_nat_irc: DCC (Direct Client to Client) support
aor@ns1:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), allow (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
53                         ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
53 (v6)                    ALLOW IN    Anywhere (v6)

aor@ns1:~$
```

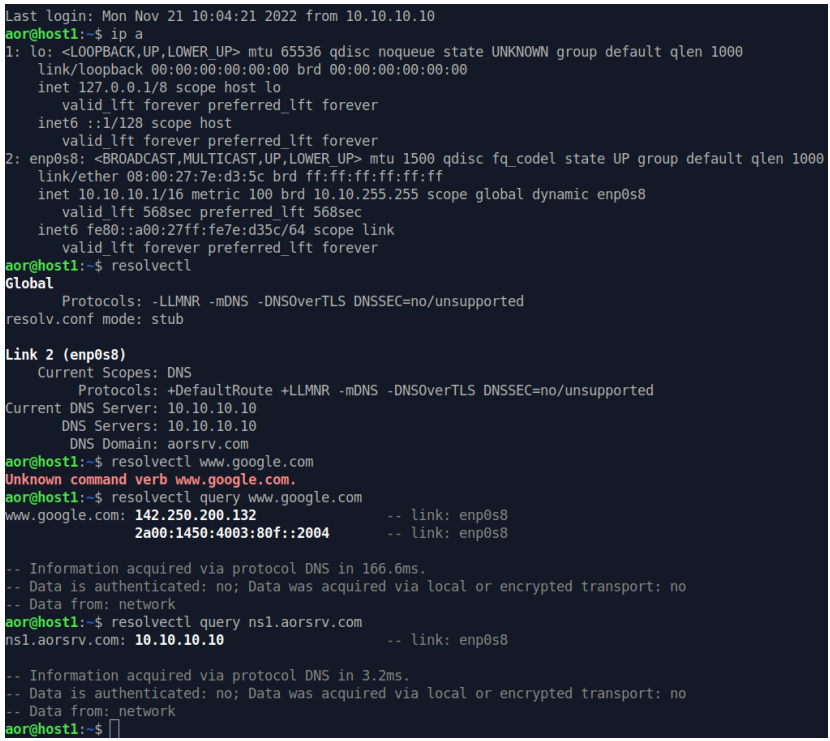
16. Ejemplo de firewall ufw.

17

```
53                         ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
53 (v6)                    ALLOW IN    Anywhere (v6)

aor@ns1:~$ ssh aor@host1.aorsrv.com
aor@host1.aorsrv.com's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-52-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of lun 21 nov 2022 10:52:55 CET

  System load:  0.39208984375     Processes:               108
  Usage of /:   54.6% of 8.02GB   Users logged in:         0
  Memory usage: 21%               IPv4 address for enp0s8: 10.10.10.1
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Mon Nov 21 10:04:21 2022 from 10.10.10.10
aor@host1:~$
```

17. Así si realizamos una conexión desde una máquina externa vía SSH a lo que sería nuestro router, que a su vez va a ser de DNS y DHCP server tenemos que entrar sin ningún tipo de problema. Igualmente podríamos hacer conexiones desde el router a cualquier entrada en la red interna.

18



18. Verificamos que todo está funcionando correctamente.

19

19. De igual forma en el DHCP server podemos hacer un dhcp-lease-list para ver que efectivamente los clientes están obteniendo las direcciones IP dentro del rango que nosotros estamos suministrando.

20

Clonar - Copiar host.

Cambiar nombre del host, cambiarlo en:

- /etc/host
- /etc/hostname

Cambiar el machine-id de la máquina, si no lo hacemos el dhcp cliente tomará la misma dirección ip para los dos hosts Ver: machine-id

```
rm -f /etc/machine-id /var/lib/dbus/machine-id
dbus-uuidgen --ensure=/etc/machine-id
dbus-uuidgen --ensure
```

Luego reiniciar la máquina

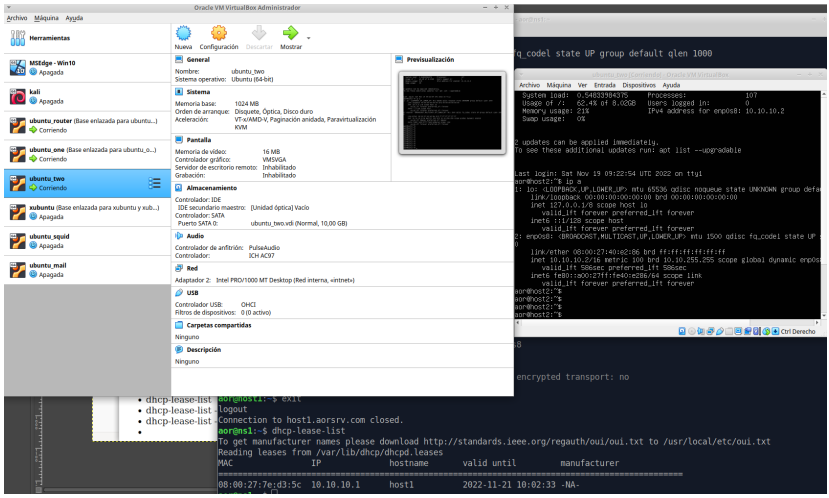
dchp

Sentencias:

- dhcp-lease-list
- dhcp-lease-list --all
- dhcp-lease-list --last

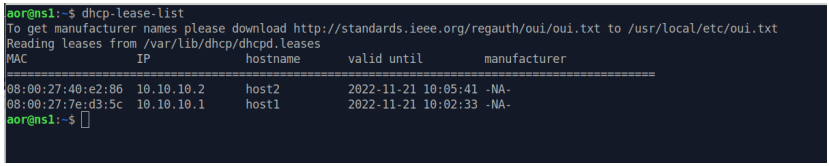
20. Podemos crear distintos clientes dentro de la misma red interna, clonando el configurado.
Tenemos que tener en cuenta posteriormente cambiarle el nombre para ello hay que modificar el host y el hostname y después posteriormente tendremos que cambiar lo que es el identificador de máquina machine-id.

21



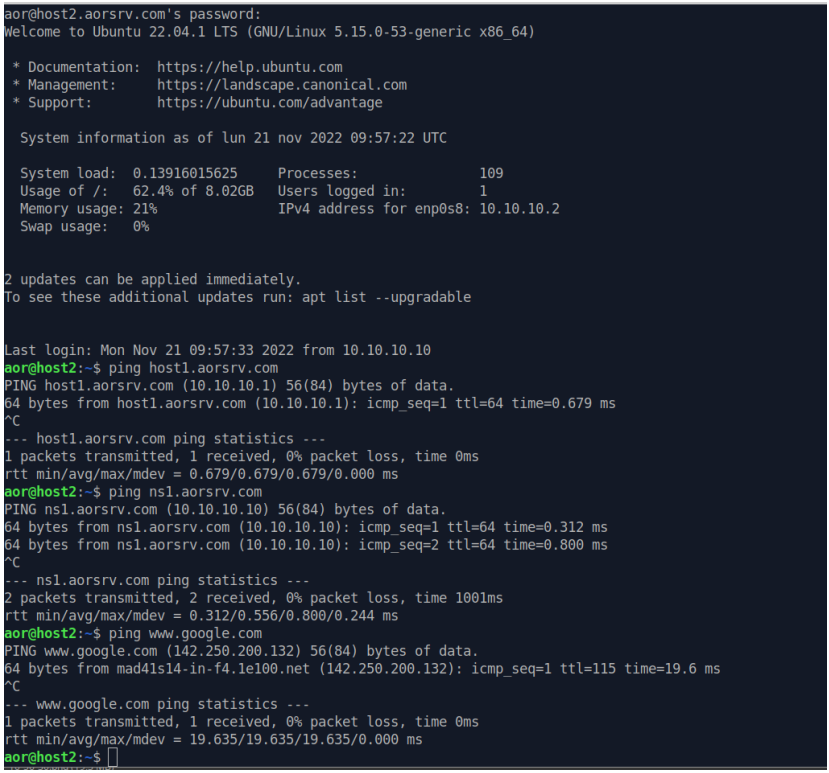
21. Se recomienda copiar elementos que van a estar dentro de la red interna para las pruebas, si tenemos problemas de espacio: enlazándolos y si no pues mejor copiarla para que no tengamos problemas con la pérdida del primero.

22



22. Cada vez que pongamos en marcha un nuevo host en la red interna debemos verificar que el DHCP server está suministrándolas de manera correcta y que los nombres son distintos.

23



23. Para cada uno de los nuevos miembros de la red que pongamos en marcha podemos hacer verificaciones de ping y dig con respecto a los otros así como el router.

24. Recuerda que el router aparte de la cuestión del firewall debe permitir un enrutamiento: tenemos que establecer net.ipv4.ip_forward = 1 en el /etc/sysctl.conf.

25. /etc/ufw/before.rules

```
#
# rules.before
#
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains:
#   ufw-before-input
#   ufw-before-output
#   ufw-before-forward
#
####AOR##########
*nat
:POSTROUTING ACCEPT [0:0]

# Forward traffic from eth1 through eth0.
-A POSTROUTING -s 10.10.10.10/16 -o enp0s3 -j MASQUERADE

# don't delete the 'COMMIT' line or these nat table rules won't be processed
COMMIT
####AOR#########
# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines


# allow all on loopback
-A ufw-before-input -i lo -j ACCEPT
-A ufw-before-output -o lo -j ACCEPT

# quickly process packets for which we already have a connection
-A ufw-before-input -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-output -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-forward -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP

# ok icmp codes for INPUT
--More--(45%)
```

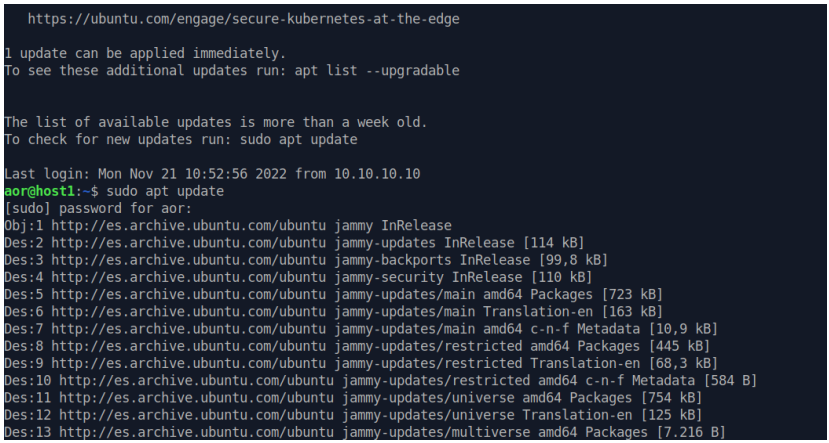
25. De igual manera tendremos que definir lo que es el NAT a nivel del firewall.

26

```
aor@ns1:~$ dhc
dhclient        dhclient-script  dhcpd           dhcp-lease-list
aor@ns1:~$ dhcp-lease-list
To get manufacturer names please download http://standards.ieee.org/regauth/oui/oui.txt to /usr/local/etc/oui.txt
Reading leases from /var/lib/dhcp/dhcpd.leases
MAC               IP              hostname       valid until         manufacturer
==============================================================================================
08:00:27:40:e2:86  10.10.10.2      host2          2022-11-21 10:05:41 -NA-
08:00:27:7e:d3:5c  10.10.10.1      host1          2022-11-21 10:02:33 -NA-
aor@ns1:~$
```

26. Verificar conexiones internas y externas, desde dentro por ejemplo tratando de actualizar el sistema.

27



Notas

Add routes de manera permanente

En netplan

```
routes:
  - to: 192.168.2.0/24
    via: 192.168.1.100
    metric: 100
  - to: 192.168.10.100
    via: 192.168.1.100
    metric: 100
```

networked

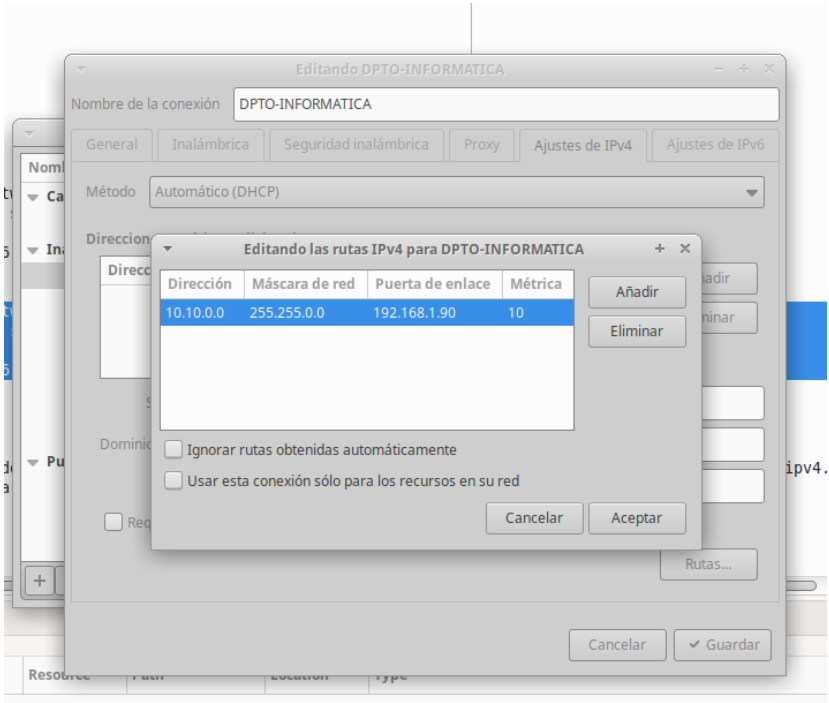
`sudo vi /etc/network/interfaces`

At the bottom of this section, add the following line:

```
up route add -net 10.10.0.0/16 gw 192.168.1.90 dev
wlp1s0
```

NetworkManager

28



28. si queremos conectar desde cualquier máquina externa al interior de la red interna para ello debemos determinar el CECU cuáles son las rutas podemos hacerlo bien si estamos en network manager o con Network configuramos lo que es la ruta hacia la red interna a través del gateway que va a ser la tarjeta de red que tenemos en nuestro servidor en la red que se encuentra en el exterior.

El resultado quedará, por ejemplo, en el fichero de configuración: `/etc/NetworkManager/system-connections/DPTO-INFORMATICA.nmconnection`

```
[ipv4]
dns=192.168.1.72;
method=auto
route1=10.10.0.0/16,192.168.1.90,10
```

Logging

```
cat /etc/bind/named.conf
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.logging";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Necesitamos crear el directorio `/var/log/named/` que vamos a usar en el siguiente fichero de configuración.

```
cat /etc/bind/named.conf.logging
logging {
    channel my_log {
        file "/var/log/named/my_log.log" versions 3 size
5m;
        severity dynamic;
        print-time yes;
    };

    category default {
        my_log;
    };
};
```

```
sudo chown bind:bind /var/log/named
sudo chmod 755 /var/log/named
sudo rm /var/log/named/my_log.log
sudo systemctl restart named.services
```

Opciones utilizadas:

channel

Define un canal de salida para los registros, donde especificas a dónde se enviarán los mensajes (por ejemplo, a un archivo, al syslog del sistema, etc.). Puedes configurar múltiples canales con diferentes configuraciones de destino

y formato.

category
: Asocia categorías específicas de eventos de BIND con uno o más canales predefinidos. Al asignar categorías a canales, controlas qué tipo de información se registra en cada destino.

versions
: Especifica el número de versiones archivadas del archivo de registro que se mantendrán antes de que las versiones más antiguas se eliminen o sobrescriban.

size
: Define el tamaño máximo que el archivo de registro puede alcanzar antes de que se realice la rotación de registros (es decir, se cree una nueva versión y se archiven las anteriores).

print-time
: Indica si se debe incluir la marca de tiempo en cada entrada del registro.

```
print-time yes;
```

print-severity
: Determina si se debe incluir el nivel de severidad (importancia) en la entrada del registro.

```
print-severity yes;
```

print-category
: Indica si se debe incluir la categoría de registro en la entrada del registro.

```
print-category yes;
```

Niveles de severidad

critical
: Eventos críticos que indican problemas serios que afectan

el funcionamiento del servicio DNS.

```
severity critical;
```

error

Errores que no son críticos pero que aún requieren atención.

```
severity error;
```

warning

Advertencias que indican situaciones potencialmente problemáticas.

```
severity warning;
```

notice

Eventos notables que no son necesariamente problemáticos, pero que pueden ser informativos.

```
severity notice;
```

info

Información general sobre la actividad del servicio DNS.

```
severity info;
```

debug

Información de depuración detallada que puede ser útil para identificar problemas específicos.

```
severity debug;
```

dynamic

Permite que el nivel de severidad se ajuste dinámicamente según la importancia del evento. Si eliges esta opción, BIND adaptará el nivel de registro en tiempo real.

```

```

```
severity dynamic;
```

Enlaces de interés

- an-introduction-to-dns-terminology-components-and-concepts

Directivas

auth-nxdomain no;

Esta directiva determina si el servidor debe responder con el bit de "Respuesta Autoritativa" (AA) cuando devuelve un error de dominio inexistente (NXDOMAIN) para zonas de las que no es maestro.
Configuración recomendada: El valor **no** es el predeterminado en versiones actuales de BIND9 y asegura que el servidor cumpla estrictamente con el estándar RFC 1035.
Mantener este valor es una medida de seguridad pasiva.
Si un servidor DNS se configura de manera insegura (por ejemplo, como un "Open Resolver" o forzando respuestas AA incorrectas), se exponen los siguientes vectores de ataque:

Reconocimiento y Revelación de Información:
Permitir respuestas detalladas a consultas de dominios inexistentes puede facilitar el **Footprinting** o reconocimiento de la red. Un atacante puede deducir la topología interna o patrones de nombres de host basándose en las respuestas del servidor, utilizando esta información para ataques dirigidos posteriores.

Riesgo de Envenenamiento de Caché (DNS Cache Poisoning):
Si un servidor no autoritativo acepta y almacena en su caché respuestas fraudulentas sobre dominios inexistentes (o existentes) sin validación suficiente, puede ser víctima de envenenamiento. Esto permitiría a un atacante redirigir el tráfico legítimo hacia servidores maliciosos.

Explotación por Fuerza Bruta (Enumeración):
Al proporcionar respuestas consistentes para nombres inexistentes, un atacante puede automatizar ataques de fuerza bruta para "mapear" cuáles son los nombres de dominio que **sí** existen por eliminación, facilitando la fase de reconocimiento de un ataque informático.

Vulneración de la Privacidad del Usuario:
: El análisis de las consultas NXDOMAIN permite a un observador externo (o a un atacante con acceso al tráfico) reconstruir patrones de navegación y comportamiento de los usuarios internos de la red, lo que representa una brecha de privacidad significativa.

Otras direcivas

allow-query
: Esta directiva especifica qué hosts o subredes están autorizados para realizar consultas al servidor DNS. Limitar las consultas a clientes específicos puede ayudar a prevenir ataques de denegación de servicio (DoS) o consultas abusivas.

allow-recursion
: Controla qué hosts o subredes pueden utilizar el servidor DNS para realizar consultas recursivas. Limitar la recursión solo a clientes confiables puede ayudar a prevenir ataques de amplificación de DNS y a reducir la carga en el servidor.

allow-transfer
: Define qué hosts o subredes pueden realizar transferencias de zona (zone transfers) desde el servidor DNS. Limitar las transferencias de zona solo a servidores autorizados puede ayudar a prevenir la divulgación no autorizada de información de zona.

recursion
: Esta directiva habilita o deshabilita la recursión en el servidor DNS. Deshabilitar la recursión en servidores que no necesitan proporcionar servicios recursivos puede ayudar a prevenir ataques de amplificación de DNS y a reducir la carga en el servidor.

listen-on
: Especifica en qué direcciones IP y puertos debe escuchar el servidor DNS. Limitar las interfaces de red en las que escucha el servidor puede ayudar a reducir la exposición a ataques.

allow-query-cache

Controla qué hosts o subredes pueden realizar consultas de caché al servidor DNS. Limitar el acceso a la caché del servidor puede ayudar a prevenir consultas abusivas o DoS.

version
: Esta directiva controla si el servidor DNS revela su versión en las respuestas a consultas. Ocultar la versión del servidor puede ayudar a evitar que los atacantes aprovechen las vulnerabilidades conocidas en versiones específicas de BIND.

allow-notify
: Define qué hosts o subredes pueden enviar notificaciones de zona al servidor DNS. Limitar las notificaciones de zona solo a servidores autorizados puede ayudar a prevenir la divulgación no autorizada de información de zona.

allow-update
: Especifica qué hosts o subredes pueden enviar actualizaciones dinámicas al servidor DNS. Limitar las actualizaciones dinámicas solo a clientes autorizados puede ayudar a prevenir cambios no autorizados en la información de zona.

query-source
: Controla desde qué dirección IP y puerto el servidor DNS envía consultas salientes. Especificar una dirección IP específica puede ayudar a evitar que el servidor sea utilizado para realizar ataques de amplificación de DNS.

max-cache-ttl
: Limita el tiempo de vida máximo de las entradas de caché en el servidor DNS. Reducir el tiempo de vida de la caché puede ayudar a mitigar los efectos de los ataques de envenenamiento de caché.

logging

Registro de consultas DNS y notificación de errores:

```
logging {
    channel query_logging {
        file "/var/log/query.log" versions 3 size 5m;
```

```
        severity debug;
        print-category yes;
        print-severity yes;
        print-time yes;
    };
    category queries { query_logging; };
    category default { query_logging; };
};
```

Además de **DNSSEC (Domain Name System Security Extensions)**, hay varios mecanismos y prácticas de seguridad que se pueden implementar para fortalecer la seguridad de un servidor DNS.

Algunos de ellos incluyen:

TSIG (Transaction Signature)
: TSIG proporciona autenticación de mensajes entre servidores DNS, lo que ayuda a prevenir ataques de suplantación de identidad y garantiza la integridad de las transferencias de zona entre servidores.

DNS Firewalling
: Implementar firewalls a nivel de DNS puede ayudar a filtrar y bloquear consultas maliciosas o sospechosas. Esto puede ayudar a prevenir ataques de denegación de servicio (DoS) y otros tipos de ataques.

Response Rate Limiting (RRL)
: RRL es una técnica que limita la tasa de respuestas que un servidor DNS puede enviar a un cliente específico. Esto ayuda a mitigar ataques de amplificación DNS al limitar la cantidad de respuestas enviadas a una IP sospechosa.

DANE (DNS-based Authentication of Named Entities)
: DANE permite a los administradores de dominios especificar qué certificados SSL/TLS son válidos para sus servidores utilizando registros DNS asegurados con DNSSEC. Esto ayuda a prevenir ataques de interceptación basados en certificados falsificados emitidos por autoridades de certificación comprometidas.

Monitorización y Registro de Logs
: Mantener registros detallados de las consultas DNS y auditar periódicamente los logs puede ayudar a detectar actividades inusuales o posibles intentos de ataque, lo que facilita la respuesta temprana ante incidentes de seguridad.

Actualizaciones Regulares
: Mantener el software del servidor DNS (como BIND,

Unbound, etc.) actualizado con los últimos parches de seguridad es fundamental para mitigar vulnerabilidades conocidas y mantener el entorno protegido contra nuevas amenazas.

Fortalecer la seguridad de un servidor DNS

Además de la directiva `version "DNS server";` en las opciones de BIND para ocultar la versión del servidor, existen otras configuraciones importantes en `named.conf.options`:

Ocultar la versión del servidor:

```
options {
    version "No disponible";
};
```

Al ocultar la versión, dificultas que los atacantes identifiquen vulnerabilidades específicas asociadas con una versión particular de BIND.

Limitar las consultas (Queries):

```
options {
    allow-query { 192.168.1.0/24; localhost; };
};
```

Permite consultas solo desde la red interna (192.168.1.0/24) y localhost. Esto evita que usuarios externos utilicen tu servidor DNS para resolver nombres, reduciendo la exposición a ataques de amplificación DNS.

Limitar las transferencias de zona (Zone Transfers):

```
options {
    allow-transfer { none; }; // O especificar
IPs de servidores secundarios de confianza
};
```

Restringe quién puede solicitar una copia completa de tus zonas DNS. Configurarlo como `none` o limitarlo a servidores secundarios de confianza evita el espionaje de la estructura de tu red (reconocimiento).

Deshabilitar la recursión para usuarios externos:

```
options {
    recursion no;
};
```

Si el servidor es puramente autoritativo (solo sirve tus propios dominios), la recursión debe desactivarse. Si necesitas recursión para usuarios internos, puedes usar vistas (views) o limitar mediante `allow-recursion`.

Limitar la recursión (si está habilitada):

```
options {
    allow-recursion { 192.168.1.0/24; localhost;
};
};
```

Permite la recursión solo a clientes de confianza. Esto previene que el servidor sea explotado como un "open resolver", una configuración peligrosa que a menudo se usa en ataques DDoS distribuidos.

DNS sobre TLS (DoT) y DNS sobre HTTPS (DoH)

Tradicionalmente, las consultas DNS se envían en texto claro, lo que significa que cualquiera en la ruta de red puede ver qué dominios estás intentando visitar. DoT y DoH solucionan esto cifrando las consultas.

- **DoT (DNS over TLS):**

- Cifra las consultas DNS utilizando el protocolo TLS (Transport Layer Security).
- Utiliza un puerto dedicado: **puerto 853**.
- Es fácil de bloquear o filtrar por los administradores de red porque utiliza un puerto específico.

- **DoH (DNS over HTTPS):**
  - Cifra las consultas DNS empaquetándolas dentro de sesiones HTTP/2 o HTTP/3 protegidas por TLS.
  - Utiliza el **puerto 443**, que es el mismo puerto utilizado para todo el tráfico web HTTPS estándar.
  - Es mucho más difícil de bloquear o filtrar, ya que las consultas DNS se mezclan visualmente con el tráfico web estándar.

Característica	DNSSEC	DoT	DoH
Propósito	Autenticidad e integridad.	Privacidad mediante cifrado.	Privacidad mediante cifrado.
Protección contra...	Spoofing, cache poisoning.	Espionaje, manipulación.	Espionaje, manipulación.
Cifrado	✗ No.	✓ Sí (TLS).	✓ Sí (HTTPS/TLS).
Protocolo usado	DNS con extensiones de firma.	DNS sobre TLS (puerto 853).	DNS sobre HTTPS (puerto 443).

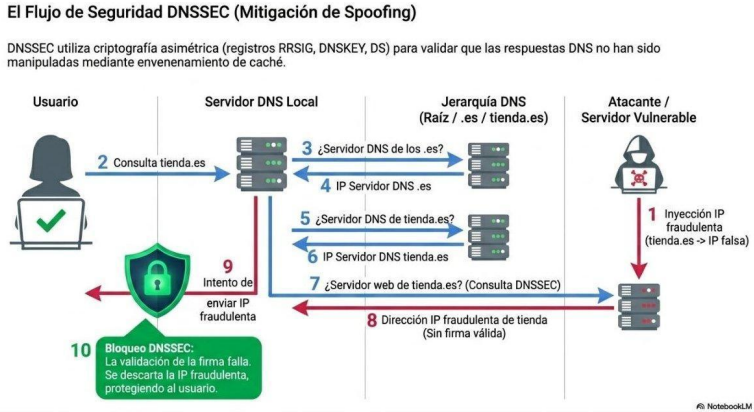
Ejemplo práctico combinando tecnologías

Sistema para usar un resolver DNS que soporta tanto DNSSEC como DoT/DoH:

- El cliente envía consultas cifradas usando DoH o DoT (protegen la privacidad del canal de comunicación).
- El resolver verifica las firmas digitales con DNSSEC (protege la integridad y autenticidad de las respuestas DNS) para asegurar que los datos recibidos no fueron alterados.
- El usuario obtiene una respuesta confiable y segura, tanto en integridad como en privacidad.

DNSSEC

Domain Name System Security Extensions es una suite de extensiones de seguridad para el protocolo DNS.
Su objetivo principal es agregar una capa adicional de seguridad al sistema de nombres de dominio (DNS), protegiendo contra ataques de envenenamiento de caché y otras formas de manipulación maliciosa de los datos de DNS. *DNSSEC*



Conceptos clave relacionados con DNSSEC:

Firmas Digitales

DNSSEC utiliza firmas digitales para autenticar la información del DNS.
Cada registro en una zona DNS firmada digitalmente lleva consigo una firma digital que puede ser verificada para asegurarse de que los datos no han sido manipulados.

Jerarquía de Claves Públicas

DNSSEC opera mediante la creación de una jerarquía de claves públicas.
En la parte superior de esta jerarquía se encuentra la "Clave de Zona" (Zone Signing Key, ZSK) que firma digitalmente todos los registros en una zona.
La Clave Maestra de Zona (Key Signing Key, KSK) se utiliza para firmar la ZSK. Estas claves se utilizan para generar

firmas digitales.

Registro DS
: La información sobre las claves utilizadas en una zona DNSSEC se envía a los servidores raíz y se propaga hacia abajo en forma de registros DS (Delegation Signer).
El registro DS se utiliza para establecer la confianza entre las zonas.

Validación de Firmas
: Cuando un cliente realiza una consulta DNS, el servidor DNS proporciona la firma digital junto con los datos.
El cliente puede validar la firma digital utilizando la clave pública de la zona, que normalmente se obtiene a través de una cadena de confianza hasta la raíz.

Respuestas de Autenticación
: Si una respuesta DNS incluye información firmada digitalmente y la firma se valida correctamente, se considera auténtica.
Si la firma no es válida, la respuesta se descarta.

NSEC y NSEC3
: DNSSEC también aborda la enumeración de zona (descubrimiento de nombres de dominio en una zona) mediante el uso de los registros NSEC (Next Secure) o NSEC3, que proporcionan pruebas criptográficas de que un nombre de dominio específico no existe en una zona.

La implementación de DNSSEC requiere acciones coordinadas a lo largo de la cadena de confianza del DNS, desde los operadores de dominios hasta los registros raíz y los servidores DNS locales.
Al implementar DNSSEC, se fortalece la integridad y autenticidad de las consultas DNS, proporcionando una capa adicional de seguridad contra ataques dirigidos a manipular o suplantar información en el sistema de nombres de dominio.

Ejemplo

- Paso 1: Instalación de BIND

```
sudo apt update
```

```
sudo apt install bind9
```

- Paso 2: Configuración de BIND para DNSSEC
  Edita el archivo de configuración de BIND:

```
sudo nano /etc/bind/named.conf.options
```

Asegúrate de que la configuración incluya opciones para DNSSEC, por ejemplo:

```
options {
    ...
   dnssec-validation auto;
    ...
};
```

Creamos el directorio keys y generamos el conjunto de claves que usaremos para nuestro dominio:

```
sudo mkdir /etc/bind/keys/
sudo dnssec-keygen -K /etc/bind/keys/ -a
NSEC3RSASHA1 -b 4096 -n ZONE -3 -f KSK aorsrv.com
sudo dnssec-keygen -K /etc/bind/keys/ -a
NSEC3RSASHA1 -b 2048 -n ZONE aorsrv.com
```

```
$ cd /etc/bind/keys
$ sudo dnssec-keygen -K /etc/bind/keys/ -a
ECDSAP256SHA256 -n ZONE aorsrv.com
Generating key pair.
Kaorsrv.com.+013+59634
$ sudo dnssec-keygen -K /etc/bind/keys/ -a
ECDSAP256SHA256 -n ZONE -f KSK aorsrv.com
Generating key pair.
Kaorsrv.com.+013+62080
```

Firmamos las claves:

```
sudo cd /etc/bind/keys
sudo dnssec-signzone -t -N INCREMENT -o aorsrv.com
-d /etc/bind -S -o aorsrv.com /etc/bind/
db.aorsrv.com
```

Resultado de la ejecución:

```
Fetching aorsrv.com/ECDSAP256SHA256/62080 (KSK)
from key repository.
Fetching aorsrv.com/ECDSAP256SHA256/59634 (ZSK)
from key repository.
Verifying the zone using the following algorithms:
- ECDSAP256SHA256
Zone fully signed:
Algorithm: ECDSAP256SHA256: KSKs: 1 active, 0
stand-by, 0 revoked
                            ZSKs: 1 active, 0
stand-by, 0 revoked
/etc/bind/db.aorsrv.com.signed
Signatures generated:                       17
Signatures retained:                         0
Signatures dropped:                          0
Signatures successfully verified:            0
Signatures unsuccessfully verified:          0
Signing time in seconds:                 0.003
Signatures per second:                4251.062
Runtime in seconds:                      0.011
```

```
sudo nano /etc/bind/named.conf.local
```

Agrega la configuración de la zona, incluyendo las claves generadas:

```
zone "aorsrv.com" IN {
    type primary;
    file "/etc/bind/db.aorsrv.com.signed";
};
```

Reinicia BIND:

```
sudo systemctl restart bind9
```

- Paso 3: Verificación y Monitoreo Verifica la Configuración de DNSSEC:

```
sudo named-checkconf
```

```
sudo named-checkzone aorsrv.com /etc/bind/
db.aorsrv.com.signed
```

Monitorea el Registro de BIND:

```
sudo tail -f /var/log/syslog | grep named
```

- Paso 4: Configuración del Cliente DNS En un cliente DNS, como un servidor DNS recursivo, no hay configuraciones adicionales necesarias.
  Los clientes modernos generalmente admiten DNSSEC de forma predeterminada.

Rotación regular de claves criptográficas

Se refiere a un proceso en el que las claves criptográficas utilizadas en DNSSEC se actualizan periódicamente con el fin de mejorar la seguridad. Estas claves se dividen generalmente en dos tipos principales:

KSK (Key Signing Key)
: Esta clave se utiliza para firmar las claves ZSK y establecer la cadena de confianza en la zona. La rotación de la KSK es un proceso crítico y generalmente se realiza con menos frecuencia que la rotación de la ZSK.

ZSK (Zone Signing Key)
: Esta clave se utiliza para firmar individualmente los registros en la zona DNS. La rotación de la ZSK es más frecuente que la rotación de la KSK y ayuda a limitar el impacto potencial de la compromisión de una clave.

Ejemplos

`dig @192.168.110.69 host1.aorsrv.com`

```
; <<>> DiG 9.18.18-0ubuntu0.23.04.1-Ubuntu <<>>
@192.168.110.69 host1.aorsrv.com
; (1 server found)
;; global options: +cmd
```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
35977
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY:
0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE:
3812dc98edd1eb4e01000000657ac4abeaf7ed960d79fa00 (good)
;; QUESTION SECTION:
;host1.aorsrv.com.  IN A

;; ANSWER SECTION:
host1.aorsrv.com. 604800 IN A 192.168.2.1

;; Query time: 0 msec
;; SERVER: 192.168.110.69#53(192.168.110.69) (UDP)
;; WHEN: Thu Dec 14 10:02:35 CET 2023
;; MSG SIZE  rcvd: 89
```

`dig @192.168.110.69 host1.aorsrv.com +dnssec`

```
; <<>> DiG 9.18.18-0ubuntu0.23.04.1-Ubuntu <<>>
@192.168.110.69 host1.aorsrv.com +dnssec
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
48296
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY:
0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
; COOKIE:
1f3c0c86d329cb1d01000000657ac4b7946c1ed64d6968a1 (good)
;; QUESTION SECTION:
;host1.aorsrv.com.  IN A

;; ANSWER SECTION:
host1.aorsrv.com. 604800 IN A 192.168.2.1
host1.aorsrv.com. 604800 IN RRSIG A 13 3 604800
20240113075345 20231214075345 59634 aorsrv.com.
4VIFAvyppd5t4zZmhZFYEfXugmZtYtOJVT/G7VPU+Jko34YRnREd1MPx
GKO1/fzHiczfQpGuVrNQQaPGcEiLMg==
```

```
;; Query time: 0 msec
;; SERVER: 192.168.110.69#53(192.168.110.69) (UDP)
;; WHEN: Thu Dec 14 10:02:47 CET 2023
;; MSG SIZE  rcvd: 195
```

Rotación claves

La rotación regular de claves criptográficas en DNSSEC puede y debería ser automatizada para garantizar la seguridad y facilitar el proceso.

1. **Scripting y Herramientas de Automatización:** Puedes utilizar scripts o herramientas de automatización para realizar el proceso de rotación de claves. Esto puede incluir la generación de nuevas claves, la actualización de las configuraciones del servidor DNS y la propagación de las claves a los servidores secundarios.
2. **Uso de Herramientas Integradas:** Algunos servidores DNS, como BIND, tienen herramientas integradas que facilitan la rotación de claves. Por ejemplo, BIND incluye utilidades como `dnssec-keygen` y `dnssec-settime` que pueden ser utilizadas en scripts de automatización.
3. **Planificación de la Rotación:** Establece un cronograma para la rotación de claves y automatiza el proceso según ese cronograma. La frecuencia de rotación puede variar, pero generalmente se realiza al menos una vez al año para las claves ZSK y con menos frecuencia para las claves KSK.
4. **Monitoreo Automático:** Implementa monitoreo automático para verificar la salud de las claves y detectar cualquier problema potencial. Esto puede incluir la verificación de fechas de vencimiento, la detección de cambios no autorizados y otros indicadores de posibles problemas de seguridad.
5. **Registro y Auditoría:** Asegúrate de mantener registros detallados de las actividades de rotación de claves. Esto es crucial para auditorías de seguridad y para entender el

historial de cambios en las claves.

6. **Pruebas Automáticas después de la Rotación:** Después de la rotación automática de claves, realiza pruebas automáticas para verificar que la zona DNS firmada con las nuevas claves se esté distribuyendo correctamente y que los clientes puedan validar la información correctamente.
7. **Actualización de Configuraciones de Zona DNS:** Automatiza la actualización de las configuraciones de la zona DNS para incluir las nuevas claves. Esto puede ser parte del script o proceso de rotación.
8. **Backup Automático:** Implementa una rutina de copia de seguridad automática de las claves antes de realizar la rotación. Esto proporciona un mecanismo de recuperación en caso de problemas durante el proceso de rotación.

Gestión de Claves y Firma de Zonas DNSSEC

> *Es fundamental diferenciar entre la **validación DNSSEC** (que el servidor ya hace al tener `dnssec-validation auto;`) y la **firma de zonas**, que es el proceso donde el administrador asegura su propio dominio.*
>
> *DNSSEC protege la integridad del sistema utilizando criptografía asimétrica para asegurar que las respuestas no han sido manipuladas. A continuación, desarrollo el apartado exhaustivo sobre la gestión de claves y el proceso de firma para tus alumnos.*

Para que un dominio sea seguro, no basta con validar las firmas de otros; el administrador debe **firmar su propia zona**. Este proceso crea una "Cadena de Confianza" que garantiza que los datos DNS provienen del origen legítimo.

1. Los Tres Pilares de DNSSEC: Registros RRSIG, DNSKEY y DS

Para entender el proceso de firma, debemos conocer estos tres

nuevos tipos de registros de recursos:

- **DNSKEY:** Contiene la **clave pública** del servidor. Es la que los clientes (resolvers) descargan para verificar las firmas de la zona.
- **RRSIG (Resource Record Signature):** Es la **firma digital** de un conjunto de registros (como un registro A o MX). Cada vez que un servidor responde, envía el registro solicitado junto con su correspondiente RRSIG para demostrar su autenticidad.
- **DS (Delegation Signer):** Es un "puntero" que se coloca en la zona del **padre** (por ejemplo, si tu zona es `aor.com`, el registro DS va en la zona `.com`). Contiene un resumen (hash) de tu clave pública, permitiendo que el mundo confíe en tu servidor porque el padre "da fe" de tu clave.

2. Tipos de Claves: KSK vs. ZSK

En una implementación profesional, se utilizan dos niveles de claves para equilibrar seguridad y mantenimiento:

1. **ZSK (Zone Signing Key):** Se usa para firmar los registros de datos de la zona (genera los RRSIG). Se recomienda rotarla con frecuencia (cada pocos meses).
2. **KSK (Key Signing Key):** Es una clave de mayor jerarquía que **solo firma la clave ZSK**. Es la clave que se vincula con el padre mediante el registro DS. Al ser la base de la confianza, se guarda con mayor celo y se rota con menos frecuencia (cada año o más).

3. Proceso Práctico de Firma (Paso a Paso)

Para un manual de BIND9, el flujo de trabajo pedagógico es el siguiente:

A. Generación de claves: Se deben generar ambos pares de claves (pública y privada) usando el algoritmo recomendado (actualmente RSASHA256 o ECDSAP256SHA256):

```
# Generar la ZSK

```

```
dnssec-keygen -a RSASHA256 -b 1024 -n ZONE aor.com

# Generar la KSK
dnssec-keygen -a RSASHA256 -b 2048 -f KSK -n ZONE
aor.com
```

B. Inclusión en el fichero de zona: Se añaden las claves públicas resultantes (archivos `.key`) al final de su archivo de zona original (`db.aor.com`):

```
$INCLUDE Kaor.com.+008+12345.key  ; Clave ZSK
$INCLUDE Kaor.com.+008+67890.key  ; Clave KSK
```

C. Firma de la zona: Se utiliza la herramienta `dnssec-signzone` para generar el archivo firmado (que tendrá la extensión `.signed`):

```
dnssec-signzone -A -3 <sal_aleatoria> -N INCREMENT -o
aor.com db.aor.com
```

*Este comando genera automáticamente todos los registros **RRSIG** para cada nombre del archivo.*

D. Publicación del registro DS: Este es el paso crítico para la seguridad externa. El comando de firma anterior genera un archivo llamado `dsset-aor.com.`. El administrador debe entregar el contenido de este archivo a su registrador de dominios para que lo publique en la zona superior. **Sin este paso, la cadena de confianza se rompe** y DNSSEC no sirve de nada.

4. Configuración en BIND9

Una vez generada la zona firmada (`db.aor.com.signed`), se debe actualizar `named.conf.local` para apuntar al nuevo archivo:

```
zone "aor.com" {
    type primary;
    file "/etc/bind/zones/db.aor.com.signed";
};
```

5. Verificación Pedagógica

Podemos usar `dig` para ver que funciona correctamente buscando específicamente los registros de seguridad:

```
dig +dnssec aor.com A
```

Si la firma es correcta, la respuesta de BIND9 incluirá una sección **ANSWER** con el registro A y una sección **RRSIG** asociada, confirmando que el dato está protegido contra ataques de suplantación o envenenamiento de caché.

Nota editorial: En versiones modernas de BIND (9.16+), existe una directiva llamada `dnssec-policy` que automatiza todo este proceso.
Sin embargo, para un manual educativo, es preferible explicar el método manual descrito arriba para que se comprenda la relación técnica entre la clave privada, la firma RRSIG y el registro DS en el padre.

Nota Técnica: Automatización Profesional con dnssec-policy

> *IMPORTANTE: De la teoría a la producción*
> *Aunque el proceso manual detallado en este manual (uso de dnssec-keygen y dnssec-signzone) es pedagógicamente esencial para entender la relación técnica entre la clave privada, la firma RRSIG y el registro DS, en entornos de producción modernos (BIND 9.16+) este método se considera propenso a errores humanos.*
> *La práctica estándar actual en la industria es el uso de la directiva dnssec-policy. Esta funcionalidad permite que el servidor BIND gestione de forma totalmente automática todo el ciclo de vida de DNSSEC: desde la generación inicial de claves (ZSK y KSK) hasta su rotación periódica y la actualización de firmas.*

> *Riesgo laboral: Un administrador que solo conozca el método manual corre el riesgo de dejar un dominio en estado "BOGUS" (inválido) si olvida realizar una rotación manual a tiempo, lo que resultaría en una caída total del servicio por errores de validación (SERVFAIL). Por tanto, una vez comprendida la base técnica, es imperativo que el alumno investigue la implementación de políticas automatizadas para su uso en servidores reales.*

Diagnóstico de Fallos DNSSEC

El **estado "BOGUS"**. Este es un punto vital para la seguridad.
Cuando DNSSEC falla (por una rotación mal hecha o un ataque real), el servidor devuelve un error SERVFAIL.
Un dominio entra en estado "Bogus" cuando la cadena de confianza se rompe
(por ejemplo, si rotaste la KSK pero olvidaste actualizar el registro DS en el registrador/padre).
Necesitamos herramientas visuales como DNSViz o DNSSEC Analyzer. Son fundamentales para ver la cadena de confianza desde la raíz (.) hasta su dominio (aor.com)

Enlaces de interés

- dnssec-guide
- dnsviz

BIND9 (named) para admitir DNS sobre TLS (DoT)

Usamos stunnel como un puente de seguridad adicional entre el servidor DNS y los clientes, para asegurar que todo el tráfico DNS esté cifrado.
Ver: BIND9 no soporta directamente la terminación de TLS

```
sudo apt-get install stunnel4
```

```
sudo openssl genpkey -algorithm RSA -out private_key.pem
-aes256
sudo openssl req -new -x509 -key private_key.pem -out
certificate.pem -days 365
```

Le quito el password para que no de errrores:

```
sudo openssl rsa -in /etc/stunnel/private_key.pem -out /
etc/stunnel/private_key_no_pass.pem
```

Creo fichero de configuración:

```
sudo touch /etc/stunnel/stunnel.conf
```

```
aor@serverA:/etc/stunnel$ cat stunnel.conf
[dns-tls]
accept = 853
connect = 127.0.0.1:53
cert = /etc/stunnel/certificate.pem
key = /etc/stunnel/private_key_no_pass.pem
```

```
sudo systemctl start stunnel4
```

```
```

```
aor@serverA:/etc/stunnel$ sudo ss -tulpn | grep 53
udp   UNCONN 0      0       172.16.5.103:53
0.0.0.0:*    users:(("named",pid=2146,fd=22))

udp   UNCONN 0      0          127.0.0.1:53
0.0.0.0:*    users:(("named",pid=2146,fd=19))

tcp   LISTEN 0      10      172.16.5.103:53
0.0.0.0:*    users:(("named",pid=2146,fd=23))

tcp   LISTEN 0      4096         0.0.0.0:853
0.0.0.0:*    users:(("stunnel4",pid=3546,fd=9))

tcp   LISTEN 0      5          127.0.0.1:953
0.0.0.0:*    users:(("named",pid=2146,fd=18))

tcp   LISTEN 0      10         127.0.0.1:53
0.0.0.0:*    users:(("named",pid=2146,fd=20))
```

```
aor@serverA:/etc/stunnel$ dig www.google.com

; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>>
www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
38022
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE:
7d8711650986bbd901000000675aba26c6c42f97a7551f3a (good)
;; QUESTION SECTION:
;www.google.com.   IN A

;; ANSWER SECTION:
www.google.com.  117 IN A 142.250.184.4

;; Query time: 35 msec
;; SERVER: 172.16.5.103#53(172.16.5.103) (UDP)
;; WHEN: Thu Dec 12 10:25:42 UTC 2024
;; MSG SIZE  rcvd: 87
```

```
aor@serverA:/etc/stunnel$ dig +tls www.google.com

; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>> +tls
www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
19925
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE:
56b9260e23131b8b01000000675aba2d714672139d83169a (good)
;; QUESTION SECTION:
;www.google.com.    IN A

;; ANSWER SECTION:
www.google.com.  110 IN A 142.250.184.4

;; Query time: 3 msec
;; SERVER: 172.16.5.103#853(172.16.5.103) (TLS)
;; WHEN: Thu Dec 12 10:25:49 UTC 2024
;; MSG SIZE  rcvd: 87
```

```
aor@serverA:/etc/stunnel$ dig +tls host1.micasa.com

; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>> +tls
host1.micasa.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
17535
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY:
0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE:
827323f3809d218001000000675abaedb1606f40f56d7208 (good)
;; QUESTION SECTION:
;host1.micasa.com.  IN A
```

```
;; ANSWER SECTION:
host1.micasa.com. 604800 IN A 172.16.5.105

;; Query time: 1 msec
;; SERVER: 172.16.5.103#853(172.16.5.103) (TLS)
;; WHEN: Thu Dec 12 10:29:01 UTC 2024
;; MSG SIZE  rcvd: 89
```

Notas

Otra forma:

- `openssl genpkey -algorithm RSA -out private_key.pem` Genera una clave privada RSA sin ninguna protección de contraseña.
- `openssl req -new -x509 -key private_key.pem -out certificate.pem -days 365` Genera un certificado autofirmado utilizando la clave privada creada previamente. El certificado tendrá una validez de 365 días (puedes cambiar el número de días según lo necesites).

Otros usos de stunnel

Stunnel puede ser utilizado para cifrar cualquier servicio que utilice conexiones TCP.

Stunnel es una herramienta flexible que puede ser utilizada para cifrar prácticamente cualquier servicio que no soporte TLS nativamente. Algunas aplicaciones comunes incluyen servicios de correo, bases de datos, FTP, y aplicaciones de acceso remoto como RDP y VNC. Además de proporcionar una capa de seguridad adicional, te permite cifrar comunicaciones que de otro modo serían inseguras, como las conexiones SMTP, Telnet, IRC, y LDAP.

Blacklist-for-Bind9

Una zona de política de respuesta o RPZ (Response Policy Zone) funciona como un "cortafuegos" a nivel de DNS que permite al administrador de la red intervenir en el proceso de resolución de nombres basándose en criterios de seguridad.
A diferencia de una zona DNS convencional que busca entregar la dirección IP real de un sitio, una zona RPZ está diseñada para que el servidor pueda "mentir" deliberadamente al cliente si detecta que este intenta acceder a un dominio peligroso, como sitios de phishing, servidores de control de malware o publicidad intrusiva.
Esta técnica se conoce comúnmente como **DNS Sinkhole**.
Cuando un usuario solicita un dominio que figura en la "lista negra" de la RPZ, el servidor DNS no devuelve la IP del atacante, sino que redirige la consulta hacia una IP segura controlada por el administrador (donde se puede mostrar una advertencia) o simplemente responde que el dominio no existe (NXDOMAIN).
De este modo, la RPZ se convierte en una herramienta de seguridad avanzada esencial para proteger a los dispositivos de la red interna, bloqueando las amenazas antes incluso de que se establezca una conexión IP con el servidor malicioso

Vamos a instalar para el uso de listas negras Grabbing-Blacklist-for-Bind9-RPZ/wiki/Fitting-Environment

```
wget https://github.com/ngadmini/Grabbing-Blacklist-for-Bind9-RPZ/releases/download/v10.4/gb4rpz-v10.4.tar.gz
mkdir gb4rpz-v10
cd gb4rpz-v10/
tar tfvz ../gb4rpz-v10.4.tar.gz
tar xfvz ../gb4rpz-v10.4.tar.gz
```

Nota seguiremos en gran parte los pasos de: readme-ov-file

```
sudo apt install dos2unix
```

```
sudo apt install faketime
sudo apt install idn
sudo apt install libnet-netmask-perl
```

Ahora ejecutamos: ./grab_http.sh

```
$ ./grab_http.sh
starting grab_http.sh v10.4 at vie 2024-12-13 10:00:04
UTC
PREPARING TASKs: check availability configuration file
                        isOK
PREPARING TASKs: check grab_http.sh is executed by non-
root privileges          isOK
PREPARING TASKs: check required debian-packages in
local-host: 172.16.5.103     isOK
PREPARING TASKs: check script-pack's property in local-
host: 172.16.5.103       isOK
PREPARING TASKs: check the actual longest of Top Level
Domains                  isOK
PREPARING TASKs: check availability of sources-urls (as
listed in grab_urls)
       url_0: dsi.ut-capitole.fr/..?../adult.tar.gz
                        isUP
       url_1: dsi.ut-capitole.fr/..?../gambling.tar.gz
                        isUP
       url_2: dsi.ut-capitole.fr/..?../phishing.tar.gz
                        isUP
       url_3: dsi.ut-capitole.fr/..?../publicite.tar.gz
                        isUP
       url_4: dsi.ut-capitole.fr/..?../redirector.tar.gz
                        isUP
       url_5: dsi.ut-capitole.fr/..?../vpn.tar.gz
                        isUP
       url_6: blocklistproject.github.io/..?../porn-
nl.txt                      isUP
       url_7: trustpositif.kominfo.go.id/..?../
domains_isp                       curl-code: 28
       url_8: pgl.yoyo.org/..?../serverlist.php?
hostformat=nohtml               isUP
       url_9: v.firebog.net/..?../Easylist.txt
                        isUP
      url_10: v.firebog.net/..?../AdguardDNS.txt
                        isUP
      url_11: s3.amazonaws.com/..?../simple_ad.txt
                        isUP
      url_12: raw.githubusercontent.com/..?../
```

```
hostnames.txt                        isUP
      url_13: urlhaus.abuse.ch/..?../rpz
                        isUP
      url_14: blocklistproject.github.io/..?../phishing-
nl.txt                    isUP
      url_15: blocklistproject.github.io/..?../malware-
nl.txt                     isUP
      url_16: blocklistproject.github.io/..?../
ransomware-nl.txt                 isUP
      url_17: raw.githubusercontent.com/..?../malware
                        isUP
      url_18: gist.githubusercontent.com/..?../raw
                        isUP
------------------------------
All URLs of sources-url(s) isUP.

PERFORMING TASKs: initiating CATEGORY of domains
       ADULT: dsi.ut-capitole.fr/..?../adult.tar.gz
                        DONE
    GAMBLING: dsi.ut-capitole.fr/..?../gambling.tar.gz
                        DONE
    PHISHING: dsi.ut-capitole.fr/..?../phishing.tar.gz
                        DONE
   PUBLICITE: dsi.ut-capitole.fr/..?../publicite.tar.gz
                        DONE
  REDIRECTOR: dsi.ut-capitole.fr/..?../redirector.tar.gz
                        DONE
         VPN: dsi.ut-capitole.fr/..?../vpn.tar.gz
                        DONE
 categorized: adult ipv4 malware publicite redirector
trust+ (6 CATEGORIES)

processing TRUST+ CATEGORY with (2) additional sources-
url(s)
      grab_1: dsi.ut-capitole.fr/..?../gambling.tar.gz
                        DONE
      grab_7: trustpositif.kominfo.go.id/..?../
domains_isp                         DONE
    reducing: porn domains and move it's to ADULT
CATEGORY                        DONE
      fixing: bads, duplicates and false entries at
TRUST+                          DONE
      moving: IP-address entries into IPV4 CATEGORY
                        DONE
    acquired: 0 entries.

processing ADULT CATEGORY with (2) additional sources-
```

```
url(s)
      grab_0: dsi.ut-capitole.fr/..?../adult.tar.gz
                        DONE
      grab_6: blocklistproject.github.io/..?../porn-
nl.txt                        DONE
      fixing: bads, duplicates and false entries at
ADULT                         DONE
      moving: IP-address entries into IPV4 CATEGORY
                        DONE
    acquired: 4.499.955 entries.

processing REDIRECTOR CATEGORY with (2) additional
sources-url(s)
      grab_4: dsi.ut-capitole.fr/..?../redirector.tar.gz
                        DONE
      grab_5: dsi.ut-capitole.fr/..?../vpn.tar.gz
                        DONE
      fixing: bads, duplicates and false entries at
REDIRECTOR                    DONE
      moving: IP-address entries into IPV4 CATEGORY
                        DONE
    acquired: 112.543 entries.

processing PUBLICITE CATEGORY with (5) additional
sources-url(s)
      grab_3: dsi.ut-capitole.fr/..?../publicite.tar.gz
                        DONE
      grab_8: pgl.yoyo.org/..?../serverlist.php?
hostformat=nohtml                DONE
      grab_9: v.firebog.net/..?../Easylist.txt
                        DONE
     grab_10: v.firebog.net/..?../AdguardDNS.txt
                        DONE
     grab_11: s3.amazonaws.com/..?../simple_ad.txt
                        DONE
      fixing: bads, duplicates and false entries at
PUBLICITE                     DONE
      moving: IP-address entries into IPV4 CATEGORY
                        DONE
    acquired: 61.768 entries.

processing MALWARE CATEGORY with (7) additional sources-
url(s)
      grab_2: dsi.ut-capitole.fr/..?../phishing.tar.gz
                        DONE
     grab_12: raw.githubusercontent.com/..?../
hostnames.txt                      DONE
```

```
     grab_13: urlhaus.abuse.ch/..?../rpz
                         DONE
     grab_14: blocklistproject.github.io/..?../phishing-
nl.txt                   DONE
     grab_15: blocklistproject.github.io/..?../malware-
nl.txt                    DONE
     grab_16: blocklistproject.github.io/..?../
ransomware-nl.txt                DONE
     grab_17: raw.githubusercontent.com/..?../malware
                         DONE
      fixing: bads, duplicates and false entries at
MALWARE                      DONE
      moving: IP-address entries into IPV4 CATEGORY
                         DONE
    acquired: 1.202.789 entries.

processing IPV4 CATEGORY with (0) additional sources-
url(s)
      fixing: bads, duplicates and false entries at IPV4
                         DONE
    acquired: 29.521 entries.

processing sources-urls (6 CATEGORIES) in summary:
       adult: 4.499.955 entries
        ipv4:    29.521 entries
     malware: 1.202.789 entries
   publicite:    61.768 entries
  redirector:   112.543 entries
      trust+:    71.787 entries
       TOTAL: 5.978.363 entries
  disk-usage:   153.503 Megabytes

[DONE] completed IN 6m:28s:117ms
[WARN] there are still remaining :
   - duplicate entries across CATEGORIES
   - sub-domains and it's parent-domain across
CATEGORIES
   - ipv4 that can be reduced to a CIDR-block
[HINT] consider continue to next TASKs
you have 4 options to proceed to the next task:
   1) pruning ipv4, duplicate & sub-domain entries
across CATEGORIES
   2) option [1] and rewriting ipv4 & domain entries to
RPZ format-entry [db.* files]
   3) option [2] and incrementing serial zones [rpz.*
files]
   4) option [3] and syncronizing latest [rpz.* and
```

```
db.*] files to rpz.warnet-ersa.net
      WARNING: remote-host rpz.warnet-ersa.net will
REBOOT due to low memory
               use 'shutdown -c' at remote-host:
rpz.warnet-ersa.net to abort
  ^C) use (Ctrl+C) to quit
please hit: (1|2|3|4) to continue OR (Ctrl+C) to quit
```

1

```
[1'st] TASK options chosen
CONTINUED to :
   1) pruning ipv4, duplicate & sub-domain entries
across CATEGORIES
PERFORMING TASKs: based on 1'st options ...

[1'st] TASKs:
starting grab_duplic.sh v10.4 at vie 2024-12-13 10:09:23
UTC
[INFO] FOUND 6 CATEGORIES: adult ipv4 malware publicite
redirector trust+
[PRUNE] duplicate entries across CATEGORIES
pruning duplicate entries based on ADULT
   STEP 0.1 = pruning duplicates ipv4 entries
SKIP
   STEP 0.2 = pruning duplicates malware entries
DONE
   STEP 0.3 = pruning duplicates publicite entries
DONE
   STEP 0.4 = pruning duplicates redirector entries
DONE
   STEP 0.5 = pruning duplicates trust+ entries
DONE
pruning duplicate entries based on IPV4
do nothing
pruning duplicate entries based on MALWARE
   STEP 2.3 = pruning duplicates publicite entries
DONE
   STEP 2.4 = pruning duplicates redirector entries
DONE
   STEP 2.5 = pruning duplicates trust+ entries
DONE
pruning duplicate entries based on PUBLICITE
   STEP 3.4 = pruning duplicates redirector entries
DONE
   STEP 3.5 = pruning duplicates trust+ entries
```

```
DONE
pruning duplicate entries based on REDIRECTOR
   STEP 4.5 = pruning duplicates trust+ entries
DONE
pruning duplicate entries based on TRUST+
do nothing

[PRUNE] IPV4 and sub-domains if parent domain exist
across CATEGORIES    DONE
[PRUNE] domains whose TLD is invalid. FOUND: 60 invalid
TLDs            DONE
[PRUNE] turn-back pruned entries to proper CATEGORIES
   turn-back pruned domains entry to ADULT category
 : 4.491.840 entries
   turn-back pruned ipv4-addresses to IPV4 category
 :    18.423 entries
   turn-back pruned domains entry to MALWARE category
 :   975.794 entries
   turn-back pruned domains entry to PUBLICITE category
 :    50.820 entries
   turn-back pruned domains entry to REDIRECTOR category
 :   110.870 entries
   turn-back pruned domains entry to TRUST+ category
 :    70.988 entries

TOTAL : 5.718.735 entries
                                                  disk-
usage :    145.63 Megabytes
[DONE] completed IN 0m:58s:173ms

TASKs based on 1'st options: DONE
bye!
```

Seguimos los pasos:

```
sudo apt policy rsync
sudo apt install rsync
sudo mkdir /etc/bind/zones-rpz
sudo cp {db,rpz}.* /etc/bind/zones-rpz/.
ls /etc/bind/zones-rpz
sudo ln -s /etc/bind/zones-rpz/rpz.* /etc/bind/
```

Cambiamos: /etc/bind/named.conf

```
sudo cat /etc/bind/named.conf
```

```
acl "bogon" { 0/8; 100.64/10; 169.254/16; 192/24;
192.0.2/24; 198.18/15; 198.51.100/24; 203.0.113/24;
224/3; 240/4;};
acl "inter" { localhost; 192.168/16; 172.16/12; 10/8; };
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
//include "/etc/bind/named.conf.logs";
include "/etc/bind/rndc.key";
controls { inet 127.0.0.1 port 953 allow { 127.0.0.1; }
keys { rndc-key; }; };
server ::/0 { bogus yes; };
```

Vamos a comentar nuestra propia configuración para ponerla en otro lugar:

```
sudo cat /etc/bind/named.conf.local
//zone "micasa.com" {
// type master;
// file "/etc/bind/zones/db.micasa.com"; # Path del
fichero de zona directa
//};
//zone "16.172.in-addr.arpa" {
//    type primary;
//    file "/etc/bind/zones/db.16.172";
//};
view "trusted" {
   match-clients {inter;};
   recursion yes;
   allow-query {inter;};
   allow-query-cache {inter;};
   allow-recursion {inter;};
   allow-transfer {none;};
   response-policy { # split in separated zone files
      zone "adultaa.domain";
      zone "adultab.domain";
      zone "adultac.domain";
      zone "adultad.domain";
      zone "adultae.domain";
      zone "adultaf.domain";
      zone "adultag.domain";
      zone "ipv4.domain";
      zone "malware.domain";
      zone "publicite.domain";
      zone "redirector.domain";
      zone "safe.domain";
      zone "trust+aa.domain";
```

```
      zone "trust+ab.domain";
      zone "trust+ac.domain";
      } break-dnssec yes qname-wait-recurse no nsip-
wait-recurse no max-policy-ttl 604800;
   include "/etc/bind/zones.rfc1918"; # we don't use
reverse-zone
   include "/etc/bind/named.conf.default-zones";
   include "/etc/bind/named.conf.trusted-zones";
   };

view "untrusted" chaos {
   match-clients {any;};
   recursion no;
   zone "." {type hint;file "/dev/null";};
   zone "bind" {type master;file "/etc/bind/
rpz.chaos";allow-query {inter;};allow-transfer
{none;};};
   };
```

```
sudo touch /etc/bind/named.conf.trusted-zones
```

```
sudo vi /etc/bind/named.conf.trusted-zones
zone "adultaa.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultaa";};
zone "adultab.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultab";};
zone "adultac.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultac";};
zone "adultad.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultad";};
zone "adultae.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultae";};
zone "adultaf.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultaf";};
zone "adultag.domain" {type master;check-names
ignore;file "/etc/bind/rpz.adultag";};
zone "ipv4.domain" {type master;check-names ignore;file
"/etc/bind/rpz.ipv4";};
zone "malware.domain" {type master;check-names
ignore;file "/etc/bind/rpz.malware";};
zone "publicite.domain" {type master;check-names
ignore;file "/etc/bind/rpz.publicite";};
zone "redirector.domain" {type master;check-names
ignore;file "/etc/bind/rpz.redirector";};
zone "safe.domain" {type master;check-names ignore;file
```

```
"/etc/bind/rpz.safe";};
zone "trust+aa.domain" {type master;check-names
ignore;file "/etc/bind/rpz.trust+aa";};
zone "trust+ab.domain" {type master;check-names
ignore;file "/etc/bind/rpz.trust+ab";};
zone "trust+ac.domain" {type master;check-names
ignore;file "/etc/bind/rpz.trust+ac";};
zone "warnet-ersa.net" {type master;file "/etc/bind/
rpz.warsa";};
```

```
 ls /etc/bind/
bind.keys  Ktlsa.+013+46295.key      named.conf.options
         rpz.adultab  rpz.ipv4        rpz.trust+ac
db.0       Ktlsa.+013+46295.private
named.conf.options.20241212  rpz.adultac  rpz.malware
 zones
db.127     named.conf
named.conf.options.old       rpz.adultad  rpz.publicite
 zones.rfc1918
db.255     named.conf.20241212       named.conf.trusted-
zones     rpz.adultae  rpz.redirector  zones-rpz
db.empty   named.conf.default-zones  rndc.key
         rpz.adultaf  rpz.trust+aa
db.local   named.conf.local          rpz.adultaa
         rpz.adultag  rpz.trust+ab
```

Nota: **SI ESTAMOS TRABAJANDO, COMO EN NUESTRO CASO, CON SUBREDES DADAS DE ALTA EN /etc/bind/zones.rfc1918 LO TENEMOS QUE CAMBIAR**

```
$ cat /etc/bind/zones.rfc1918
zone "10.in-addr.arpa"      { type master; file "/etc/
bind/db.empty"; };
// #AOR#
//zone "16.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
// #AOR#
zone "17.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "18.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "19.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "20.172.in-addr.arpa"  { type master; file "/etc/
```

```
bind/db.empty"; };
zone "21.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "22.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "23.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "24.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "25.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "26.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "27.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "28.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "29.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "30.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };
zone "31.172.in-addr.arpa"  { type master; file "/etc/
bind/db.empty"; };

zone "168.192.in-addr.arpa" { type master; file "/etc/
bind/db.empty"; };
```

```
$ sudo cat /etc/bind/named.conf.default-zones
// prime the server with knowledge of the root servers
zone "." {
 type hint;
 file "/usr/share/dns/root.hints";
};

// be authoritative for the localhost forward and
reverse zones, and for
// broadcast zones as per RFC 1912

zone "localhost" {
 type master;
 file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
 type master;
 file "/etc/bind/db.127";
```

```
};

zone "0.in-addr.arpa" {
 type master;
 file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
 type master;
 file "/etc/bind/db.255";
};

zone "micasa.com" {
 type master;
 file "/etc/bind/zones/db.micasa.com"; # Path del
fichero de zona directa
};

zone "16.172.in-addr.arpa" {
    type primary;
    file "/etc/bind/zones/db.16.172";
};
```

NOTA: MEJOR SACAR NUESTRAS CONFIGURACIONES PROPIAS DE /etc/bind/zones.rfc1918 y /etc/bind/named.conf.default-zones

mDNS (Multicast DNS)

mDNS es un protocolo de resolución de nombres que permite a los dispositivos en una red local descubrir y conectarse entre sí utilizando multicast en lugar de depender de un servidor DNS central.
En sistemas operativos como Ubuntu, mDNS está integrado y se utiliza para facilitar la detección de servicios y dispositivos en la red local.

Concepto: El "Grito" Multicast en mDNS

Para entender mDNS, los alumnos primero deben recordar cómo funciona el **DNS clásico**:

> *En el DNS normal, cuando un equipo quiere saber la IP de `google.com`, va de forma ordenada e individual (Unicast) a un "servidor central" (como el `8.8.8.8` o el router de casa) y le pregunta en privado. El servidor, que tiene una base de datos, le responde en privado.*

¿Pero qué pasa si montamos una red improvisada para un evento, un aula de prácticas aislada, o estamos en casa y el router no tiene internet ni servidor DNS configurado? Si no hay un "coordinador central", los equipos se quedan ciegos y mudos. No pueden traducirse los nombres.

Ahí es donde mDNS cambia el juego con la filosofía del **Grito por Multicast**:

1. **La Dirección del Megáfono (`224.0.0.251`):** En lugar de hablar con un servidor, el equipo que busca a otro coge un "megáfono virtual". Envía un paquete UDP al puerto `5353` con destino a la IP `224.0.0.251`. Esta dirección IP es

especial (Multicast): el router y los switches de la red saben que **cualquier paquete enviado ahí debe ser replicado y entregado a absolutamente todos** los dispositivos conectados al mismo segmento de red.
2. **La Pregunta Abierta:** El cliente lanza su pregunta al aire: *"¿Quién se llama `aorsrv.local`? Pasadme vuestra IP"*. Todos los equipos de la sala reciben la pregunta y la leen.
3. **La Respuesta del Aludido:** Las máquinas que **no** se llaman así ignoran el paquete. Pero el equipo que efectivamente se llama `aorsrv.local` da un paso al frente y responde (también gritando por multicast o respondiendo directamente al emisor): *"¡Soy yo! Mi IP es `192.168.1.20`"*. El cliente anota la IP y ya puede iniciar la conexión.

Script de Comprobación de Red (`comprobar_mdns.sh`)

Verifica si tu sistema está preparado para "gritar y escuchar" en la red local. Realiza diagnósticos a bajo nivel (comprueba rutas multicast, sockets abiertos y conectividad en el puerto `5353`).

```
#!/bin/bash
# comprobar_mdns.sh

VERDE='\033[0;32m'
AZUL='\033[0;34m'
AMARILLO='\033[1;33m'
ROJO='\033[0;31m'
NC='\033[0m'

echo -e "${AZUL}
=========================================================
${NC}"
echo -e "${AZUL}    AUDITORÍA Y COMPROBACIÓN DE ENTORNO
mDNS/UDP      ${NC}"
echo -e "${AZUL}
=========================================================
${NC}"

# 1. COMPROBACIÓN DE RUTA MULTICAST EN EL KERNEL
echo -e "\n${AMARILLO}[1] Verificando soporte Multicast
en la tabla de rutas...${NC}"
# El kernel de Linux debe tener una ruta para el rango
```

```
224.0.0.0
if ip route | grep -q "224.0.0.0"; then
    echo -e "${VERDE}✓ OK: El sistema sabe cómo enrutar
paquetes Multicast (rango 224.0.0.0/4).${NC}"
else
    echo -e "${AMARILLO}! NOTA: No hay una ruta
explícita para 224.0.0.0, el kernel usará la ruta por
defecto.${NC}"
fi

# 2. VERIFICACIÓN DEL PUERTO Abierto (SOCKET)
echo -e "\n${AMARILLO}[2] Buscando procesos escuchando
en el puerto sagrado UDP 5353...${NC}"
# Usamos ss (sucesor de netstat) para buscar quién
retiene el puerto de mDNS
PROCESO_MDNS=$(ss -ulpn sport = :5353 2>/dev/null | grep
-v "State")

if [ -n "$PROCESO_MDNS" ]; then
    echo -e "${VERDE}✓ OK: Hay un demonio escuchando en
el puerto 5353/UDP.${NC}"
    echo -e "${AZUL}Detalle del socket:${NC}\n
$PROCESO_MDNS"
else
    echo -e "${ROJO}✗ ALERTA: Nadie está escuchando en
el puerto 5353. ¿Avahi está apagado?${NC}"
fi

# 3. VERIFICACIÓN DE FIREWALL LOCAL (UFW)
echo -e "\n${AMARILLO}[3] Comprobando reglas del
Firewall local (UFW)...${NC}"
if sudo ufw status | grep -q "Status: active"; then
    echo -e "${AMARILLO}! El Firewall UFW está ACTIVO.
${NC}"
    echo "Asegúrate de que permita el tráfico entrante/
saliente en el puerto UDP 5353."
    echo "Reglas actuales relacionadas con 5353:"
    sudo ufw status verbose | grep "5353" || echo "  [i]
No hay reglas específicas para el 5353. Podría estar
bloqueado."
else
    echo -e "${VERDE}✓ El Firewall UFW está INACTIVO. No
hay bloqueos locales para mDNS.${NC}"
fi

# 4. PRUEBA DE "GRITO" EN VIVO (Ping Multicast)
echo -e "\n${AMARILLO}[4] Lanzando un 'grito' de prueba
```

```
a la IP Multicast 224.0.0.251...${NC}"
echo "Enviando 3 pings de broadcasting. Los equipos con
mDNS activo deberían responder."
echo -e
"${AZUL}------------------------------------------------------
${NC}"

# Hacemos un ping especial con el flag -t para forzar
TTL local (no salta routers) y -b para permitir
broadcast
ping -c 3 -t 1 -b 224.0.0.251

echo -e
"${AZUL}------------------------------------------------------
${NC}"
echo -e "${AMARILLO}Análisis para la clase:${NC}"
echo "Cualquier IP (aparte de la tuya) que haya
respondido a ese ping es un"
echo "equipo 'chivato' en la LAN que tiene el protocolo
mDNS/Avahi habilitado."

echo -e "\n${VERDE}
========================================================
${NC}"
echo -e "${VERDE}            Diagnóstico de Red
Terminado              ${NC}"
echo -e "${VERDE}
========================================================
${NC}\n"
```

Ejecutamos

```
$ ./mDNS_comprobar.sh
========================================================
    AUDITORÍA Y COMPROBACIÓN DE ENTORNO mDNS/UDP
========================================================

[1] Verificando soporte Multicast en la tabla de
rutas...
! NOTA: No hay una ruta explícita para 224.0.0.0, el
kernel usará la ruta por defecto.

[2] Buscando procesos escuchando en el puerto sagrado
UDP 5353...
✓ OK: Hay un demonio escuchando en el puerto 5353/UDP.
Detalle del socket:
```

```
UNCONN 0      0            0.0.0.0:5353      0.0.0.0:*

UNCONN 0      0               [::]:5353         [::]:*


[3] Comprobando reglas del Firewall local (UFW)...
[sudo] contraseña para aor:
✓ El Firewall UFW está INACTIVO. No hay bloqueos locales
para mDNS.

[4] Lanzando un 'grito' de prueba a la IP Multicast
224.0.0.251...
Enviando 3 pings de broadcasting. Los equipos con mDNS
activo deberían responder.
------------------------------------------------------
PING 224.0.0.251 (224.0.0.251) 56(84) bytes of data.

--- 224.0.0.251 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss,
time 2039ms

------------------------------------------------------
Análisis para la clase:
Cualquier IP (aparte de la tuya) que haya respondido a
ese ping es un
equipo 'chivato' en la LAN que tiene el protocolo mDNS/
Avahi habilitado.

======================================================
            Diagnóstico de Red Terminado
======================================================
```

Análisis

Esta petición `PING 224.0.0.251` hará que respondan todos los equipos que tengan mDNS/Avahi habilitado

Desactivar mDNS

Deshabilitar completamente mDNS puede afectar la capacidad de los dispositivos para descubrir servicios y recursos en la red local.

- Desactivar el servicio Avahi (implementación de mDNS en Linux):
    - Detener el servicio y su socket (para evitar que se reactive solo):

```
sudo systemctl stop avahi-daemon avahi-daemon.socket
```

    - Deshabilitar el servicio y su socket en el arranque:

```
sudo systemctl disable avahi-daemon avahi-daemon.socket
```

- Filtrar paquetes mDNS con iptables:
  Insertamos la regla al principio de la cadena (flag -I) para asegurar el bloqueo del puerto UDP 5353:

```
sudo iptables -I INPUT -p udp --dport 5353 -j DROP
```

- Bloquear el servicio mDNS en el cortafuegos UFW: Si estás utilizando UFW (Uncomplicated Firewall), puedes bloquear el tráfico mDNS con el siguiente comando:

```
sudo ufw deny 5353/udp
```

Guía para desactivar mDNS, DNS-SD y LLMNR

Seguridad

ES COMPLETAMENTE INSEGURO SOLO EN LANs MUY CONTROLADAS

DNS-SD (Domain Name System Service Discovery)

Zeroconf (Zero Configuration Networking) es un conjunto de tecnologías que permite la configuración automática de redes y servicios sin la necesidad de intervención manual.
DNS-SD (Domain Name System Service Discovery) es una parte de Zeroconf que facilita descubrir servicios en una red local.

> *Porque mDNS solo te da IPs a partir de nombres. Pero, ¿cómo sabe tu ordenador que en esa IP hay un servidor web, una impresora o un Chromecast? Ahí entra DNS-SD (DNS Service Discovery)*

¿Qué es DNS-SD?
: DNS-SD es una extensión del protocolo DNS que permite a los dispositivos en una red anunciarse y descubrir servicios disponibles localmente. Proporciona un método estándar para que los dispositivos publiquen y descubran servicios, como impresoras, cámaras, servidores, etc., sin necesidad de configuración manual.

Funcionamiento
: Los servicios anuncian su presencia en la red mediante registros DNS específicos.
Los clientes pueden realizar consultas DNS-SD para descubrir servicios disponibles y obtener información sobre ellos.

Mitigación de Carga en la Red:

Controlar Anuncios de Servicios
: Algunos dispositivos y sistemas permiten configurar qué servicios se anuncian. Revisa la configuración de tus dispositivos para desactivar la publicidad de servicios innecesarios.

Segmentación de Red

Utiliza VLANs (Virtual LANs) para segmentar la red y limitar la visibilidad de los anuncios de servicios a segmentos específicos.

Filtrado de Anuncios

Algunos routers y firewalls permiten filtrar anuncios de servicios basándose en ciertos criterios.
Configura reglas para permitir o bloquear anuncios según tus necesidades.

```
sudo iptables -A INPUT -p udp --dport 5353 -j
DROP
```

o

```
sudo ufw deny in on <interfaz_de_red> to any port
5353
```

Control de Tráfico mDNS

Algunos routers avanzados ofrecen opciones para controlar el tráfico mDNS. Puedes limitar la propagación de mensajes mDNS más allá de ciertos límites.

Desactivar Servicios no Necesarios

Desactiva servicios de red que no necesitas. Algunos dispositivos y sistemas tienen configuraciones para desactivar completamente el soporte de Zeroconf o mDNS.

Uso de Subdominios

En entornos más grandes, considera la posibilidad de utilizar subdominios específicos para servicios DNS-SD para reducir la carga en el dominio principal.

Práctica de laboratorio

En vez de usar un DNS server real utilizaremos un Shell Script que simula las respuestas que daría un servidor DNS-SD ante las consultas del alumno
El siguiente lo ejecutamos en un server:

```
```

```
#!/bin/bash
# servidor_dns_sd.sh

VERDE='\033[0;32m'
AZUL='\033[0;34m'
AMARILLO='\033[1;33m'
NC='\033[0m'

PUERTO=10053

echo -e "${AZUL}
============================================================
${NC}"
echo -e "${AZUL}     SERVIDOR DE SIMULACIÓN DNS-SD PARA
EL AULA       ${NC}"
echo -e "${AZUL}
============================================================
${NC}"
echo -e "${AMARILLO}[i] Escuchando consultas DNS
didácticas en el puerto $PUERTO...${NC}"
echo "Deja esta ventana abierta y dile a tus alumnos que
hagan las consultas."

# Bucle infinito para responder simulaciones usando
netcat (nc)
while true; do
    # Capturamos la petición (aunque respondemos de
forma estática para la práctica)
    nc -l -u -p $PUERTO -w 1 < /dev/null > /dev/null

    # Este script simplemente sirve de soporte visual.
    # Para la práctica real usaremos un truco de mock
con comandos individuales o archivos locales.
    sleep 0.5
done
```

Ejecutamos server

```
$ ./dns_sd_simulado.sh
============================================================
     SERVIDOR DE SIMULACIÓN DNS-SD PARA EL AULA
============================================================
[i] Escuchando consultas DNS didácticas en el puerto
10053...
Deja esta ventana abierta y dile a tus alumnos que hagan
```

```
las consultas.
```

Shell cliente

Simulamos también los clientes mediante el siguiente script:

```
#!/bin/bash
# auditar_dns_sd.sh

VERDE='\033[0;32m'
AZUL='\033[0;34m'
AMARILLO='\033[1;33m'
ROJO='\033[0;31m'
CYAN='\033[0;36m'
NC='\033[0m'

clear
echo -e "${AZUL}
============================================================
${NC}"
echo -e "${AZUL}      LABORATORIO PRÁCTICO:
COMPRENDIENDO LAS TRIPAS DE DNS-SD     ${NC}"
echo -e "${AZUL}
============================================================
${NC}"
echo -e "DNS-SD no es magia; utiliza registros DNS
estándar (PTR, SRV, TXT)."
echo -e "Vamos a simular que consultamos a un servidor
DNS para descubrir servicios.\n"

# PASO 1: REGISTRO PTR
echo -e "${AMARILLO}[PASO 1] ¿Qué servicios de impresión
(_printer._tcp) hay en la red?${NC}"
echo -e "En DNS-SD, se usa un registro ${CYAN}PTR${NC}
apuntando al tipo de servicio para listarlos."
echo -e "Presiona ENTER para simular la consulta:
${VERDE}dig PTR _printer._tcp.local${NC}"
read -r
echo -e "${AZUL}--- RESPUESTA DNS (REGISTRO PTR) ---
${NC}"
echo -e ";; ANSWER SECTION:"
echo -e "_printer._tcp.local.    86400   IN  PTR
${VERDE}Impresora_Laser_Despacho._printer._tcp.local.
${NC}"
```

```
echo -e "-----------------------------------\n"
echo -e "💡 ${CYAN}Explicación para la pizarra:${NC} El
PTR nos ha dado el \"Nombre Humano\" del servicio."
echo -e "Sabemos que existe 'Impresora_Laser_Despacho',
pero no sabemos ni su IP, ni su puerto, ni sus datos.\n"

# PASO 2: REGISTRO SRV
echo -e "${AMARILLO}[PASO 2] ¿En qué puerto y máquina
está esa impresora exactamente?${NC}"
echo -e "Ahora que tenemos el nombre, DNS-SD consulta el
registro ${CYAN}SRV${NC} de ese servicio específico."
echo -e "Presiona ENTER para simular la consulta:
${VERDE}dig SRV
Impresora_Laser_Despacho._printer._tcp.local${NC}"
read -r
echo -e "${AZUL}--- RESPUESTA DNS (REGISTRO SRV) ---
${NC}"
echo -e ";; ANSWER SECTION:"
echo -e "Impresora_Laser_Despacho._printer._tcp.local.
120 IN SRV 0 0 ${ROJO}9100${NC} ${VERDE}servidor-
impresion.local.${NC}"
echo -e "-----------------------------------\n"
echo -e "💡 ${CYAN}Explicación para la pizarra:${NC} El
registro SRV nos acaba de revelar dos datos vitales:"
echo -e "1. El puerto de destino (en este caso, el
estándar de impresión: ${ROJO}9100${NC})."
echo -e "2. El nombre del host real que la aloja
(${VERDE}servidor-impresion.local.${NC}).\n"

# PASO 3: REGISTRO TXT (Los Metadatos)
echo -e "${AMARILLO}[PASO 3] ¿Qué propiedades o
características tiene este servicio?${NC}"
echo -e "Aquí viene la clave de tu pregunta. DNS-SD usa
el registro ${CYAN}TXT${NC} para meter metadatos"
echo -e "en formato clave=valor (como rutas, modelos, si
requiere contraseña, etc.)."
echo -e "Presiona ENTER para simular la consulta:
${VERDE}dig TXT
Impresora_Laser_Despacho._printer._tcp.local${NC}"
read -r
echo -e "${AZUL}--- RESPUESTA DNS (REGISTRO TXT) ---
${NC}"
echo -e ";; ANSWER SECTION:"
echo -e "Impresora_Laser_Despacho._printer._tcp.local.
120 IN TXT ${VERDE}\"txtvers=1\" \"pdl=application/
postscript\" \"model=HP_LaserJet_400\" \"path=/printers/
laser_aula\" \"admin=poveda@school.com\"${NC}"
```

```
echo -e "-----------------------------------\n"
echo -e "🔔 ${CYAN}Explicación para la pizarra:${NC}
¡Ahí están los metadatos!"
echo -e "Gracias al registro ${CYAN}TXT${NC}, nuestro
ordenador sabe, antes de conectarse, que la impresora"
echo -e "es una HP, habla PostScript, se gestiona en la
ruta ${VERDE}/printers/laser_aula${NC} y el
administrador eres tú."

echo -e "\n${VERDE}
================================================================
${NC}"
echo -e "   ¡Fin de la práctica! Los alumnos han visto
las tripas de DNS-SD  "
echo -e "${VERDE}
================================================================
${NC}\n"
```

Ejecutamos

```
================================================================
      LABORATORIO PRÁCTICO: COMPRENDIENDO LAS TRIPAS DE
DNS-SD
================================================================
DNS-SD no es magia; utiliza registros DNS estándar (PTR,
SRV, TXT).
Vamos a simular que consultamos a un servidor DNS para
descubrir servicios.

[PASO 1] ¿Qué servicios de impresión (_printer._tcp) hay
en la red?
En DNS-SD, se usa un registro PTR apuntando al tipo de
servicio para listarlos.
Presiona ENTER para simular la consulta: dig PTR
_printer._tcp.local

--- RESPUESTA DNS (REGISTRO PTR) ---
;; ANSWER SECTION:
_printer._tcp.local.    86400   IN  PTR
Impresora_Laser_Despacho._printer._tcp.local.
-----------------------------------

🔔 Explicación para la pizarra: El PTR nos ha dado el
"Nombre Humano" del servicio.
Sabemos que existe 'Impresora_Laser_Despacho', pero no
```

```
sabemos ni su IP, ni su puerto, ni sus datos.

[PASO 2] ¿En qué puerto y máquina está esa impresora
exactamente?
Ahora que tenemos el nombre, DNS-SD consulta el registro
SRV de ese servicio específico.
Presiona ENTER para simular la consulta: dig SRV
Impresora_Laser_Despacho._printer._tcp.local

--- RESPUESTA DNS (REGISTRO SRV) ---
;; ANSWER SECTION:
Impresora_Laser_Despacho._printer._tcp.local. 120 IN SRV
0 0 9100 servidor-impresion.local.
-----------------------------------

📡 Explicación para la pizarra: El registro SRV nos
acaba de revelar dos datos vitales:
1. El puerto de destino (en este caso, el estándar de
impresión: 9100).
2. El nombre del host real que la aloja (servidor-
impresion.local.).

[PASO 3] ¿Qué propiedades o características tiene este
servicio?
Aquí viene la clave de tu pregunta. DNS-SD usa el
registro TXT para meter metadatos
en formato clave=valor (como rutas, modelos, si requiere
contraseña, etc.).
Presiona ENTER para simular la consulta: dig TXT
Impresora_Laser_Despacho._printer._tcp.local

--- RESPUESTA DNS (REGISTRO TXT) ---
;; ANSWER SECTION:
Impresora_Laser_Despacho._printer._tcp.local. 120 IN TXT
"txtvers=1" "pdl=application/postscript"
"model=HP_LaserJet_400" "path=/printers/laser_aula"
"admin=poveda@school.com"
-----------------------------------

📡 Explicación para la pizarra: ¡Ahí están los
metadatos!
Gracias al registro TXT, nuestro ordenador sabe, antes
de conectarse, que la impresora
es una HP, habla PostScript, se gestiona en la ruta /
printers/laser_aula y el administrador eres tú.

============================================================
```

```
   ¡Fin de la práctica! Los alumnos han visto las tripas
de DNS-SD
================================================================
```

Práctica con Dnsmasq

Con dnsmasq podríamos usar:

Configuración

```
#
================================================================
# CONFIGURACIÓN DE PRÁCTICAS: DESCUBRIMIENTO DE
SERVICIOS (DNS-SD)
#
================================================================

# 1. Registro PTR (El Catálogo)
# Sintaxis: txt-record=<nombre>,<contenido> (Usamos txt-
record en dnsmasq para forzar PTRs de texto)
# Pregunta: ¿Qué impresoras hay? -> Respuesta:
Impresora_Laser_Despacho
ptr-
record=_printer._tcp.local,Impresora_Laser_Despacho._printer._tc

# 2. Registro SRV (La Localización)
# Sintaxis: srv-
host=<servicio>.<tipo>.<dominio>,<host_destino>,<puerto>,<priori
# Pregunta: ¿Dónde está esa impresora? -> Respuesta: En
el puerto 9100 de servidor-impresion.local
srv-
host=Impresora_Laser_Despacho._printer._tcp.local,servidor-
impresion.local,9100,0,0

# 3. Registro TXT (Los Metadatos)
# Sintaxis: txt-record=<nombre>,<metadatos separados por
comas o en bloque>
# Pregunta: ¿Qué propiedades tiene? -> Respuesta: Los
atributos clave=valor
```

```
txt-
record=Impresora_Laser_Despacho._printer._tcp.local,"txtvers=1",
postscript","model=HP_LaserJet_400","path=/printers/
laser_aula","admin=poveda@school.com"

# 4. Registro A (Resolución de IP clásica para cerrar el
ciclo)
# Asociamos el host del registro SRV a una IP real del
laboratorio
host-record=servidor-impresion.local,192.168.1.150
```

Comandos de auditoría

```
# Paso 1: Descubrir qué servicios de impresión existen
dig @192.168.1.100 PTR _printer._tcp.local +short

# Paso 2: Averiguar el puerto y la máquina que lo aloja
dig @192.168.1.100 SRV
Impresora_Laser_Despacho._printer._tcp.local

# Paso 3: Extraer los metadatos de configuración
dig @192.168.1.100 TXT
Impresora_Laser_Despacho._printer._tcp.local +short
```

Enlaces de interés

Desactivar i26_desactivar_mdns_dns-sd_llmnr.xhtml

Seguridad

El escenario inseguro: DNS-SD sobre mDNS (Avahi / Apple)

Es 100% inseguro

El escenario seguro: DNS-SD sobre DNS Clásico (Wide-Area DNS-SD)

DNS-SD también se puede configurar en un servidor DNS tradicional de Internet (como acabas de hacer con dnsmasq o con servidores como BIND9).
Esto se conoce como Wide-Area DNS-SD (Descubrimiento de servicios en redes amplias).

En este escenario, DNS-SD sí puede ser seguro porque puede aprovechar las tecnologías de protección del DNS de toda la vida:

- **DNSSEC (Firmas Digitales)**: Se pueden firmar criptográficamente los registros PTR, SRV y TXT. Si un atacante intenta suplantar la impresora del aula e inyecta un registro falso, el ordenador del alumno comprobará la firma digital, verá que no coincide y rechazará la conexión.
- **TSIG (Transferencia Segura)**: Solo los administradores autenticados con una clave secreta pueden modificar o añadir nuevos servicios al servidor DNS.

Cuadro comparativo

Característica	Entorno Doméstico / Aula (Inseguro)	Entorno Corporativo / Seguro
Mecanismo	mDNS + DNS-SD (**Avahi**)	DNS Clásico + DNS-SD (**Wide-Area**)
Transporte	Grito por Multicast (`UDP 5353`)	Consultas dirigidas al Servidor DNS (`UDP/TCP 53`)
Control de publicación	Ninguno (Cualquiera publica)	**TSIG** (Solo dispositivos autorizados)
Validación de respuestas	Ninguna (El más rápido gana)	**DNSSEC** (Respuestas firmadas digitalmente)
Ámbito	Solo la misma sección de cable/WiFi	Puede cruzar routers y sedes de la empresa

Avahi es una implementación de código abierto del protocolo de descubrimiento de servicios Zeroconf, que permite a los dispositivos en una red local descubrir y comunicarse entre sí sin la necesidad de configuración manual.

Avahi proporciona servicios como:

- La resolución de nombres y
- La asignación automática de direcciones IP

avahi-daemon

Este es el demonio principal de Avahi que gestiona la publicación y descubrimiento de servicios en la red.

```
systemctl status avahi-daemon
```

Proporciona una interfaz para que los servicios anuncien su disponibilidad y para que otros dispositivos descubran esos servicios automáticamente.

avahi-dnsconfd

Este componente se utiliza para integrar dinámicamente los servicios descubiertos por Avahi en el sistema de resolución de nombres..
Por ejemplo, puede configurar automáticamente el sistema para que use servicios DNS locales anunciados por Avahi.

avahi-browse y avahi-publish

Estas son herramientas de línea de comandos que permiten a los usuarios explorar y publicar servicios en la red, respectivamente..
avahi-browse muestra los servicios disponibles, mientras que avahi-publish permite la publicación manual de servicios.

Para examinar los servicios y equipos que Avahi ha detectado de forma dinámica, el comando más potente es avahi-browse.

- Ver todos los servicios activos en tiempo real:

```
avahi-browse -a -r -t
```

  - -a (all): Busca todos los tipos de servicios disponibles.
    - -r (resolve): Resuelve automáticamente los nombres a direcciones IP y puertos.
    - -t (terminate): Muestra la foto actual de la red y sale (si lo omites, se queda escuchando en segundo plano nuevos cambios).
  - Ver solo los dominios y equipos (sin servicios individuales):

    ```
    avahi-browse -d local _http._tcp
    ```

    (Cambia _http._tcp por el servicio específico que busques, como _ssh._tcp o _printer._sub._http._tcp).

  Publicar un servicio temporalmente (avahi-publish)

  Puedes simular o inyectar un servicio en la red DNS-SD directamente desde la terminal sin modificar archivos de configuración. Muy útil para pruebas. Publicar un servidor web ficticio:

  ```
  avahi-publish -s "Mi Web de Pruebas"
  _http._tcp 8080 "path=/demo"
  ```

  (Cualquier otro equipo en la red que use Avahi verá instantáneamente este servicio en el puerto 8080).

avahi-resolve

Esta herramienta se utiliza para resolver nombres de host y direcciones IP utilizando Avahi..

Facilita la resolución de nombres para servicios anunciados en la red.
Emular el comportamiento de un comando nslookup o dig pero utilizando mDNS:

- Traducir un nombre .local a IP:

```
avahi-resolve -n mi-servidor.local
```

- Resolución inversa (de IP a nombre .local)

```
avahi-resolve -a 192.168.1.50
```

Avahi-autoipd
Es específicamente el componente encargado de la asignación automática de direcciones IP
Cuando está en funcionamiento, puede asignar direcciones IP automáticamente a dispositivos en una red local que no hayan obtenido una dirección IP a través de un servidor DHCP.
Esto se logra utilizando el protocolo de dirección IP automática (APIPA) y permite que los dispositivos se comuniquen en la red local sin necesidad de una configuración manual de direcciones IP.

avahi-discover
Descubrir servicios de forma gráfica
Si estás en un entorno de escritorio y prefieres no leer texto plano en la terminal, puedes instalar y ejecutar el navegador visual
Abre una ventana que categoriza por árbol todos los servicios, IPs, TXT records y puertos de tu LAN.

Descubrimiento de servicios en redes locales

Estos componentes trabajan juntos para proporcionar una solución de descubrimiento de servicios en redes locales, permitiendo que los dispositivos descubran y utilicen servicios sin necesidad de configuración manual extensa.

En una red es necesario definir el rango de direcciones IP o la subred en la que estás trabajando.
La asignación de direcciones IP se realiza dentro de un rango de direcciones específico, y este rango debe estar configurado de manera coherente en toda la red para que los dispositivos puedan comunicarse correctamente.

Cuando utilizas servicios como Avahi-autoipd, que se encargan de la asignación automática de direcciones IP en ausencia de un servidor DHCP, es fundamental tener en cuenta el rango de direcciones IP disponible para esa asignación automática. Este rango suele ser una parte específica de la red y se define de antemano.

> *Avahi no gestiona directamente la asignación de direcciones IP o la configuración de subredes.*
> *Su función principal se centra en el descubrimiento de servicios en la red.*

Para gestionar la asignación de direcciones IP, especialmente cuando se utiliza Avahi-autoipd para la asignación automática de direcciones IP, se necesita tener en cuenta algunos aspectos adicionales:

Configuración de Avahi-autoipd

NOs aseguramos de que Avahi-autoipd esté configurado correctamente.
Puedes ajustar la configuración en su archivo de configuración, que a menudo se encuentra en /etc/avahi/avahi-autoipd.conf.
Este archivo puede contener opciones relacionadas con el rango de direcciones IP que se utilizará para la asignación automática.
Por ejemplo:

```
[server]
allow-interfaces=eth0
```

Configuración de la Red

Además de Avahi, necesitamos configurar la red en sí misma. Definimos la subred y el rango de direcciones IP en los dispositivos de red que admiten la asignación automática de direcciones IP.

Comprobación de Conflictos

Ten en cuenta que la asignación automática de direcciones IP puede generar conflictos si no se gestiona adecuadamente. Asegura que el rango de direcciones IP utilizado por Avahi-autoipd no entre en conflicto con otras direcciones IP estáticas o asignadas por DHCP en tu red.

Ip dináminas

En un entorno donde no hay direcciones IP fijas ni servidor DHCP, Avahi-autoipd se encargará de asignar direcciones IP automáticamente a los dispositivos que lo necesiten.

```
sudo apt-get install avahi-autoipd
```

Configuración de Avahi-autoipd: `/etc/avahi/avahi-autoipd.conf` Aquí puedes especificar las interfaces en las que Avahi-autoipd debería estar activo.

```
[server]
allow-interfaces=eth0
```

En este caso, eth0 es la interfaz de red en la que Avahi-autoipd estará activo. Ajusta según tus necesidades.

```
sudo service avahi-autoipd restart   # Ubuntu/Debian
```

O usando systemd:

```
sudo systemctl restart avahi-autoipd
```

Avahi-autoipd ahora asignará automáticamente direcciones IP en el rango 169.254.x.x a los dispositivos en la interfaz especificada que no hayan obtenido una dirección IP de otro método.

Un ejemplo práctico

```
#!/bin/bash

# Colores para que los alumnos identifiquen mejor cada
fase
VERDE='\033[0;32m'
AZUL='\033[0;34m'
AMARILLO='\033[1;33m'
ROJO='\033[0;31m'
NC='\033[0m' # No Color

echo -e "${AZUL}
============================================================
${NC}"
echo -e "${AZUL}   SIMULACIÓN PRÁCTICA DE AVAHI Y MDNS
PARA ALUMNOS   ${NC}"
echo -e "${AZUL}
============================================================
${NC}"

# 0. VERIFICACIÓN DE REQUISITOS
echo -e "\n${AMARILLO}[Fase 0] Verificando entorno...
${NC}"
if ! command -v avahi-browse &> /dev/null; then
    echo -e "${ROJO}Error: Las herramientas de Avahi no
están instaladas.${NC}"
    echo "Dile a tus alumnos que ejecuten: sudo apt
update && sudo apt install avahi-utils"
    exit 1
fi
echo -e "${VERDE}✓ Todo listo. Avahi está instalado.
${NC}"
sleep 1

# 1. PUBLICACIÓN (INYECTAR SERVICIO FICTICIO)
echo -e "\n${AMARILLO}[Fase 1] Publicando un servicio
web ficticio en segundo plano...${NC}"
echo "Comando: avahi-publish -s \"Web de Practicas Aula
\" _http._tcp 8080 \"path=/demo\""

# Publicamos en segundo plano (&) y guardamos el PID
para poder borrarlo luego
avahi-publish -s "Web de Practicas Aula" _http._tcp 8080
"path=/demo" &> /dev/null &
```

```
PID_SERVICIO=$!

echo -e "${VERDE}✓ Servicio publicado con éxito (PID
temporal: $PID_SERVICIO).${NC}"
sleep 2 # Esperamos a que mDNS propague el servicio en
la LAN

# 2. DESCUBRIMIENTO (AVAHI-BROWSE)
echo -e "\n${AMARILLO}[Fase 2] Buscando servicios HTTP
activos en la red (Modo Foto/Terminar)...${NC}"
echo "Comando: avahi-browse -d local _http._tcp -t -r"
echo -e
"${AZUL}----------------------------------------------------
${NC}"
# Buscamos específicamente el servicio HTTP resolviendo
nombres (-r) y cerrando al terminar (-t)
avahi-browse -d local _http._tcp -t -r
echo -e
"${AZUL}----------------------------------------------------
${NC}"
sleep 2

# 3. RESOLUCIÓN DE NOMBRES (AVAHI-RESOLVE)
# Obtenemos el hostname actual de la máquina para la
prueba de resolución
HOSTNAME_LOCAL=$(hostname).local

echo -e "\n${AMARILLO}[Fase 3] Simulando un 'nslookup'
mDNS para resolver el equipo local...${NC}"
echo "Comando: avahi-resolve -n $HOSTNAME_LOCAL"
echo -e
"${AZUL}----------------------------------------------------
${NC}"
avahi-resolve -n "$HOSTNAME_LOCAL"
echo -e
"${AZUL}----------------------------------------------------
${NC}"
sleep 2

# 4. LIMPIEZA Y ELIMINACIÓN
echo -e "\n${AMARILLO}[Fase 4] Finalizando la simulación
y eliminando el servicio...${NC}"
echo "Matando el proceso del servicio ficticio (PID:
$PID_SERVICIO)..."

# Matamos el proceso para simular que el servidor web
"se ha apagado"
```

```
kill $PID_SERVICIO 2> /dev/null

echo -e "${VERDE}✓ Servicio eliminado.${NC}"
sleep 1

# 5. VERIFICACIÓN POST-ELIMINACIÓN
echo -e "\n${AMARILLO}[Fase 5] Verificando que el
servicio ya NO existe en la red...${NC}"
echo -e
"${AZUL}----------------------------------------------------
${NC}"
# Volvemos a escanear. Ya no debería aparecer "Web de
Practicas Aula"
avahi-browse -d local _http._tcp -t
echo -e
"${AZUL}----------------------------------------------------
${NC}"

echo -e "\n${VERDE}
========================================================
${NC}"
echo -e "${VERDE}   ¡Simulación completada con éxito
para el aula!    ${NC}"
echo -e "${VERDE}
========================================================
${NC}\n"
```

Resultado ejecución

```
aor@aorsrv:~/apuntes/ServiciosRedInternet$ ./
avahi_mdns_ejemplo.sh plo.sh
========================================================
   SIMULACIÓN PRÁCTICA DE AVAHI Y MDNS PARA ALUMNOS
========================================================

[Fase 0] Verificando entorno...
✓ Todo listo. Avahi está instalado.

[Fase 1] Publicando un servicio web ficticio en segundo
plano...
Comando: avahi-publish -s "Web de Practicas Aula"
_http._tcp 8080 "path=/demo"
✓ Servicio publicado con éxito (PID temporal: 6513).

[Fase 2] Buscando servicios HTTP activos en la red (Modo
```

```
Foto/Terminar)...
Comando: avahi-browse -d local _http._tcp -t -r
-----------------------------------------------------------
+ br-b283e07138da IPv4 Web de Practicas Aula
            Web Site             local
+ docker0 IPv4 Web de Practicas Aula
    Web Site             local
+ br-1f690d26aa59 IPv4 Web de Practicas Aula
            Web Site             local
+ virbr0 IPv4 Web de Practicas Aula
   Web Site             local
+ wlp7s0 IPv6 Web de Practicas Aula
   Web Site             local
+ wlp7s0 IPv4 Web de Practicas Aula
   Web Site             local
+     lo IPv4 Web de Practicas Aula
   Web Site             local
= br-b283e07138da IPv4 Web de Practicas Aula
            Web Site             local
   hostname = [aorsrv.local]
   address = [172.18.0.1]
   port = [8080]
   txt = ["path=/demo"]
= docker0 IPv4 Web de Practicas Aula
    Web Site             local
   hostname = [aorsrv.local]
   address = [172.17.0.1]
   port = [8080]
   txt = ["path=/demo"]
= br-1f690d26aa59 IPv4 Web de Practicas Aula
            Web Site             local
   hostname = [aorsrv.local]
   address = [172.19.0.1]
   port = [8080]
   txt = ["path=/demo"]
= virbr0 IPv4 Web de Practicas Aula
   Web Site             local
   hostname = [aorsrv.local]
   address = [192.168.122.1]
   port = [8080]
   txt = ["path=/demo"]
= wlp7s0 IPv6 Web de Practicas Aula
   Web Site             local
   hostname = [aorsrv.local]
   address = [fe80::7a01:caa2:b34e:e0af]
   port = [8080]
   txt = ["path=/demo"]
```

```
= wlp7s0 IPv4 Web de Practicas Aula
   Web Site             local
   hostname = [aorsrv.local]
   address = [192.168.1.20]
   port = [8080]
   txt = ["path=/demo"]
=     lo IPv4 Web de Practicas Aula
   Web Site             local
   hostname = [aorsrv.local]
   address = [127.0.0.1]
   port = [8080]
   txt = ["path=/demo"]
------------------------------------------------------

[Fase 3] Simulando un 'nslookup' mDNS para resolver el
equipo local...
Comando: avahi-resolve -n aorsrv.local
------------------------------------------------------
aorsrv.local 172.18.0.1
------------------------------------------------------

[Fase 4] Finalizando la simulación y eliminando el
servicio...
Matando el proceso del servicio ficticio (PID: 6513)...
✓ Servicio eliminado.

[Fase 5] Verificando que el servicio ya NO existe en la
red...
------------------------------------------------------
------------------------------------------------------

======================================================
   ¡Simulación completada con éxito para el aula!
======================================================
```

> *La razón por la que resuelve a 172.18.0.1 (la IP de la interfaz bridge, normalmente de Docker o una red virtual) en lugar de la 192.168.x.x de la tarjeta Ethernet física se debe al orden de prioridad y selección de interfaces que tiene Avahi por defecto.*

Orden de uso:

- Avahi asocia un único Hostname a la máquina: Para Avahi, tu máquina se llama aorsrv.local en toda la red, globalmente.

Pero esa máquina tiene múltiples personalidades (múltiples IPs: la de la Ethernet, la de la red Wifi, la del Bridge virtual, etc.).

- El orden de las interfaces en el kernel: Cuando ejecutas avahi-resolve -n aorsrv.local, el demonio de Avahi busca en su tabla interna de interfaces. Por lo general, las interfaces virtuales tipo bridge (como br-X, docker0, etc.) se registran en el sistema o se inicializan de una forma que quedan en la parte superior de la lista de prioridades de Avahi.
- La primera respuesta gana: Al pedirle una IP para el nombre de host, Avahi responde con la primera dirección IP válida que tiene asociada a ese nombre en su lista interna. Al estar el bridge primero, te escupe la 172.18.0.1 y se queda tan tranquilo, ignorando que para ti la importante es la de la Ethernet.

Podemos:

```
sudo vi /etc/avahi/avahi-daemon.conf

[server]
# Pon aquí solo tu interfaz física (o separa por comas
si usas eth0 y wlan0)
allow-interfaces=enp3s0

sudo systemctl restart avahi-daemon
```

Observaciones

Cuando ejecutamos avahi-browse, es completamente normal (y correcto) que veamos el mismo servicio repetido varias veces, una fila por cada interfaz de red (como eth0, wlan0, docker0, interfaces virtuales veth, o el bucle local lo).

Esto ocurre por tres razones fundamentales de cómo funciona mDNS/Zeroconf:

1. mDNS funciona por "Multicast" (Difusión)

El protocolo mDNS no sabe de antemano dónde están los otros

equipos, así que envía la información "a ciegas" a una dirección IP especial de multicast (`224.0.0.251` en IPv4).

Por seguridad y diseño de redes, **el tráfico multicast no salta de una interfaz de red a otra automáticamente**. Por lo tanto, para asegurarse de que *cualquier* equipo conectado a *cualquier* red de tu máquina pueda ver el servicio, Avahi decide **escuchar y anunciar el servicio en absolutamente todas las interfaces activas** del sistema de forma independiente.

2. Interfaces Virtuales y Contenedores

Si tenemos instalado Docker, VirtualBox o servicios de virtualización en Ubuntu, tendrán interfaces virtuales. Avahi las trata como si fueran tarjetas de red físicas reales. Si creas un servicio, Avahi piensa: *"Oye, quizás un contenedor de Docker o una máquina virtual quiere conectarse a este servicio"*, así que también lo publica ahí.

3. Diferenciación de IPv4 e IPv6

Además de ver el servicio por cada interfaz, a menudo lo verás duplicado como **IPv4** y **IPv6** para la misma interfaz. Avahi publica en ambos protocolos en paralelo.

Seguridad

ES COMPLETAMENTE INSEGURO SOLO EN LANs MUY CONTROLADAS

LLMNR (Link-Local Multicast Name Resolution)

LLMNR es otro protocolo de resolución de nombres utilizado en redes locales.
Al igual que mDNS y DNS-SD, LLMNR se utiliza para resolver nombres de host a direcciones IP en la misma red local sin depender de un servidor DNS central.

¿Qué es LLMNR?
: LLMNR permite que los dispositivos en una red local resuelvan nombres de host a direcciones IP sin necesidad de un servidor DNS central. Funciona enviando solicitudes de resolución de nombres a través de multicast en la red local.

Funcionamiento
: Cuando un dispositivo en la red necesita resolver un nombre de host, envía una solicitud LLMNR multicast. Otros dispositivos en la misma red pueden responder con la dirección IP asociada al nombre de host.

Mitigación de Carga en la Red:

Desactivar LLMNR
: En entornos donde LLMNR no es necesario, se puede desactivar en los sistemas operativos que lo admiten (por ejemplo, Windows).
Esto evitará que los dispositivos envíen o respondan a solicitudes LLMNR.

Filtrado de Tráfico
: Algunos routers y firewalls permiten filtrar tráfico LLMNR. Configura reglas para permitir o bloquear el tráfico según tus necesidades.

```
sudo iptables -A INPUT -p udp --dport 5355 -j
DROP
```

o

```
sudo ufw deny in on <interfaz_de_red> to any port
5355
```

Controlar Anuncios de Servicios

Algunos sistemas operativos pueden utilizar LLMNR para anunciar servicios locales.
Si es posible, desactiva la publicidad de servicios innecesarios.

Uso de DNS en Redes Corporativas

En entornos empresariales, donde se utiliza un servidor DNS centralizado, se puede desactivar LLMNR en los dispositivos y depender completamente del servidor DNS.

Configuración en Dispositivos de Red

En algunos routers y switches, puedes configurar opciones específicas para manejar o filtrar el tráfico LLMNR.

Educación del Usuario

Educa a los usuarios sobre la seguridad y los riesgos asociados con la resolución de nombres local.
Fomenta el uso de prácticas seguras de resolución de nombres, especialmente en entornos más grandes.

- Desactivar mDNS y DNS-SD en Linux:
    - Desactivar mDNS:
      En sistemas basados en systemd (como Ubuntu), puedes desactivar el servicio avahi-daemon, que implementa mDNS:

      ```
      sudo systemctl stop avahi-daemon
      sudo systemctl disable avahi-daemon
      ```

    - Desactivar DNS-SD:
      Para desactivar la publicación de servicios mediante DNS-SD, puedes detener y desactivar el servicio relacionado:

      ```
      sudo systemctl stop avahi-daemon
      sudo systemctl disable avahi-daemon
      ```

- Desactivar LLMNR en Linux:
    - Desactivar LLMNR en systemd-resolved:
      En sistemas que utilizan systemd-resolved, puedes desactivar la resolución LLMNR editando el archivo de configuración:

      ```
      sudo nano /etc/systemd/resolved.conf
      ```

    - Agrega o modifica la línea LLMNR=no y reinicia el servicio:

      ```
      sudo systemctl restart systemd-resolved
      ```

- Desactivar mDNS y LLMNR en Windows:

- Desactivar mDNS y DNS-SD:
  - En Windows, el servicio responsable de mDNS y DNS-SD es "Servicio de anuncio de dispositivos UPnP". Puedes desactivarlo a través de la interfaz gráfica o la línea de comandos.
  - Interfaz gráfica:
    Ve a "Panel de control" > "Herramientas administrativas" > "Servicios".
    Busca "Servicio de anuncio de dispositivos UPnP", haz clic derecho y selecciona "Propiedades". Luego, establece el tipo de inicio en "Desactivado" y detén el servicio.
  - Línea de comandos:

```
Stop-Service -Name "SSDPSRV"
Set-Service -Name "SSDPSRV" -StartupType
Disabled
```

- Desactivar LLMNR: Puedes desactivar LLMNR a través de la configuración de políticas de grupo:

  1. Ejecuta gpedit.msc para abrir el Editor de directivas de grupo local.
  2. Navega a "Configuración del equipo" > "Plantillas administrativas" > "Red" > "DNS del cliente".
  3. Habilita la política "Desactivar resolución LLMNR" y establece la opción en "Activado".

Table of Contents

1. Prólogo
    1. Bloques
    2. DNS. Estructura del manual
2. DNS
    1. Lección: El Sistema de Nombres de Dominio (DNS) - La Guía Telefónica de Internet
    2. La Jerarquía del DNS: De lo Específico a lo General
    3. Terminología
    4. ¿Cómo trabaja?
    5. DNS básico
        1. Consultas recursivas
        2. /etc/resolv.conf
        3. Tipos de servidores
        4. Host ubuntu resuelve ip
        5. DNS Records
3. DNS. Productos
    1. DNS servers
4. DNS Comandos
    1. Host
    2. Whois
    3. resolvectl
    4. nslookup
    5. Dig
5. systemd-resolved
    1. DNS sobre TLS (DoT):
    2. DNS sobre HTTPS (DoH):

6. Bind9 junto con systemd-resolved
    1. Cohabitación sin loop
    2. Comprobaciones
7. DNS. Tcpdump
    1. Análisis de Tráfico DNS con tcpdump
8. DNSmasq
    1. DNSmasq
    2. DHCP Server con DNSmasq
    3. Reservas de IP (Static Leases)
    4. DNSmasq como enrutador de publicidad (Bloqueo de publicidad)
        1. ¿Cómo funciona?
    5. Configuración red del servidor ejemplo
9. DNSmasq. DHCP
    1. dnsmasq y DHCP
    2. Integración DNS-DHCP Automática
    3. Ventajas de la Integración DNS-DHCP en Dnsmasq
10. DNS
    1. master-primary
    2. Servidor DNS Master-Slave con Bind9
    3. NS1
    4. Zonas
    5. NS2
    6. Seguridad
        1. Directivas
        2. DNSSEC
        3. Recursividad sin forwarders
11. DNS. RNDC

    1. RNDC
        1. Requisitos para usar rndc
12. DNS ejemplo
    1. Ficheros ejemplo
    2. DNS Verifica
        1. Verifica puerto
13. DNS. Apuntes
    1. DNS. Forwarders y recursión
        1. Sin forwarders
    2. Acerca de la recursión
14. DDNS
    1. DNS Dinámico (DDNS)
    2. DNS
    3. DCHP
    4. Notas
        1. Add routes de manera permanente
15. DNS logging
    1. Logging
    2. Enlaces de interés
16. DNS Seguridad
    1. Directivas
        1. auth-nxdomain no;
        2. Otras direcivas
        3. logging
17. DNS. Seguridad avanzada

    1. Fortalecer la seguridad de un servidor DNS
    2. DNS sobre TLS (DoT) y DNS sobre HTTPS (DoH)

18. DNSSEC

    1. DNSSEC
    2. Ejemplo
    3. Ejemplos
    4. Rotación claves
    5. Gestión de Claves y Firma de Zonas DNSSEC
    6. Nota Técnica: Automatización Profesional con dnssec-policy
    7. Diagnóstico de Fallos DNSSEC
    8. Enlaces de interés

19. DNS sobre TLS

    1. BIND9 (named) para admitir DNS sobre TLS (DoT)

20. Blacklist

    1. Blacklist-for-Bind9

21. mDNS

    1. mDNS (Multicast DNS)
    2. Desactivar mDNS
    3. Seguridad

22. DNS-SD

    1. DNS-SD (Domain Name System Service Discovery)
    2. Práctica de laboratorio
    3. Práctica con Dnsmasq
    4. Seguridad

23. Avahi

    1. Ip dináminas
    2. Un ejemplo práctico
    3. Seguridad

24. LLMNR

1. LLMNR (Link-Local Multicast Name Resolution)

25. Desactivar mDNS, DNS-SD y LLMNR

Landmarks

1. Cover